UNIVERSIDAD DE BURGOS

TESIS DOCTORAL

Optimización de procesos industriales con técnicas de Minería de Datos: mantenimiento de aerogeneradores y fabricación con tecnologías láser

Pedro SANTOS GONZÁLEZ

1-07-2015

Directores:

Dr. Andrés BUSTILLO IGLESIAS

Dr. Jesús MAUDES RAEDO

La investigación realizada en esta tesis doctoral ha sido parcialmente subvencionada por el Ministerio de Economía y Competitividad, proyecto TIN-2011-24046.

Agradecimientos

En primer lugar quisiera agradecer a mis directores de Tesis, los doctores Andrés Bustillo Iglesias y Jesús Maudes Raedo, su generosa ayuda en la elaboración de esta Tesis, y sobre todo, en el proceso de aprendizaje que supone la realización de un Trabajo de Investigación.

Durante estos años he convivido con otros compañeros del área de Lenguajes y Sistemas Informáticos, a los que quiero agradecer su disposición, especialmente a Raúl Marticorena Sánchez, por el tiempo que me ha dedicado y sus consejos sobre programación. Gracias también a Juan José Rodríguez Díez y César García Osorio por las ideas que nos han dado para la investigación y por su apoyo durante todo este período.

Este Trabajo de Investigación no hubiera sido posible sin los conjuntos de datos utilizados para el análisis. Por ello, agradezco a los grupos de investigación colaboradores la generosidad que han tenido al compartir los datos con nosotros. Gracias a Luisa Fernanda Villa y Aníbal Reñones de CARTIF por los datos para el diagnóstico de fallos en aerogeneradores, a Daniel Teixidor y Joaquim Ciurana de la Universitat de Girona por su experimentación sobre el fresado de microcavidades, y a Aitzol Lamikiz y Eneko Ukar de la Universidad del País Vasco por los datos para el análisis del pulido superficial.

Finalmente, quiero mostrar mi agradecimiento a la Universidad de Burgos, donde he realizado los estudios de Doctorado, y que me ha proporcionado junto con la Universidad de Córdoba los medios para poder realizar la investigación. Agradezco también a Nicolás García Pedrajas que me haya permitido realizar gran parte de los experimentos del trabajo de investigación en el computador de la Universidad de Córdoba.

Resumen

El desarrollo de la Minería de Datos abre el camino a la optimización a ciertos procesos industriales. Tal es el caso de los procesos analizados en la presente Tesis Doctoral: el diagnóstico de fallos en aerogeneradores y la fabricación de piezas metálicas de geometría compleja mediante tecnologías láser. El objetivo de este estudio es aumentar la eficiencia de dichos procesos industriales, mejorando el procedimiento de validación experimental de estudios anteriores, en los que no se ha usado validación cruzada estratificada ni se han tenido en cuenta algunas particularidades de los problemas analizados, como el desequilibrio entre fallos y aciertos para el diagnóstico de fallos en aerogeneradores, y la cantidad limitada de datos en el caso de fabricación de piezas metálicas de geometría compleja.

Para el diagnóstico automático de fallos en aerogeneradores, se identifica la técnica de clasificación más adecuada para relacionar las medidas de vibraciones con el tipo de fallo. Además, se estudia la influencia del desequilibrio en los datos en la precisión de los clasificadores y se define la métrica más adecuada para evaluar la precisión de los clasificadores.

Para la fabricación de piezas metálicas de geometría compleja, se estima la técnica de clasificación más adecuada para predecir la calidad superficial obtenida con pulido superficial láser, así como la técnica de regresión más adecuada para predecir los errores en los distintos requerimientos geométricos de piezas obtenidas mediante microfresado 3D láser.

Palabras clave: Minería de Datos, optimización de procesos, diagnóstico de fallos, aerogeneradores, fabricación láser, pulido superficial láser, micro-mecanizado 3d láser, clasificación desequilibrada multiclase, métricas

Abstract

The development of Data Mining opens the door to the optimization of certain industrial processes. Such is the case of the processes analyzed in this present PhD: wind turbine fault diagnosis and the manufacturing of metallic pieces of complex geometry using laser technology. The objective of this study is to increase the efficiency of those industrial processes, improving the experimental validation procedures of previous studies, which neither used stratified cross validation nor took into account some particular characteristics of the problems under analysis, such as imbalance between the number of failures and correct situations for wind turbine fault diagnosis, and the limited amount of data in the case of the manufacturing of metallic pieces of complex geometry.

With regard to automatic wind turbine fault diagnosis, the most suitable classification technique to relate the measures of vibration levels with the type of failure is identified. Besides, the influence of the imbalance between the number of failures and correct situations on classifier accuracy is studied, and the most suitable metric to evaluate classifier performance is defined.

In the case of the manufacturing of metallic pieces of complex geometry, the most suitable classification technique to forecast the surface quality obtained with laser polishing was estimated, as well as the best regression technique to predict the errors in several geometric requirements of pieces manufactured with laser 3D micro-milling.

Keywords: Data Mining, process optimization, fault diagnosis, wind turbines, laser manufacturing, laser polishing, 3d laser micro-milling, multi-class imbalance classification, metrics

Índice general

Capítulo 1

Introducción

La presente tesis se encuadra dentro de la disciplina conocida como *Minería de Datos*, y, más específicamente, es un trabajo sobre las *Aplicaciones Industriales de las técnicas de Reconocimiento de Patrones*. En los últimos años ha crecido el interés en emplear la Minería de Datos en un contexto industrial, a causa de tres razones. En primer lugar, las empresas disponen de un volumen de información cada vez mayor, debido a un aumento considerable de los sistemas de monitorización de máquinas y a la mejora de los dispositivos de almacenamiento de información. En segundo lugar, en ciertos procesos industriales hay decisiones que se toman siguiendo el criterio de un experto basándose en su experiencia previa y que podrían automatizarse utilizando técnicas de Minería de Datos. Finalmente, se han desarrollado nuevos procesos industriales de alto valor añadido en los que hay muy poca experiencia anterior y la generación de experimentos resulta muy costosa, por lo que la extracción de la máxima información posible de los experimentos realizados es fundamental.

El desarrollo de la Minería de Datos abre el camino a la optimización a ciertos procesos industriales para los cuales aún no están definidas los condiciones de operación óptimas. Tal es el caso tanto del diagnóstico de fallos en aerogeneradores como de la fabricación de piezas metálicas de geometría compleja mediante tecnologías láser, si bien por razones muy distintas. En el caso de los aerogeneradores, la falta de optimización reside en que el diagnóstico depende de un análisis de vibraciones muy complejo y que requiere de la asistencia de un técnico experto, mientras que en el caso de la fabricación de piezas metálicas de geométrica compleja la dificultad para optimizar reside en la gran complejidad de los

modelos físicos de los procesos que tienen lugar y en la falta de conocimiento que ligue las variables físicas que dominan estos procesos con las variables que realmente pueden modificarse bajo condiciones industriales.

Respecto a estos dos problemas industriales, cabe resaltar que ya existen estudios anteriores en los que se utilizan técnicas de Minería de Datos para abordarlos. Sin embargo, en dichos trabajos las validaciones experimentales presentan ciertas limitaciones que se pretenden subsanar. En primer lugar, la estimación de la precisión de las predicciones no fue analizada en base a lo que se conoce como *validación cruzada estratificada*. Por ello, dichas estimaciones podrían ser solamente válidas para los datos analizados, no siendo posible garantizar la precisión de los modelos frente a otros datos.

En segundo lugar, a la hora de desarrollar las técnicas de Reconocimiento de Patrones, es necesario tener en cuenta que los entornos reales de producción imponen ciertas restricciones sobre los datos de partida:

- En la fabricación de piezas metálicas de geometría compleja, se dispone de pocos ejemplos para elaborar el modelo, debido a limitaciones económicas. Los ensayos son destructivos, por lo que es deseable encontrar aquellos sistemas que sean capaces de obtener predicciones precisas incluso a partir de pocos datos.

- En los parques eólicos, las situaciones en las que un aerogenerador falla son muy poco frecuentes en comparación con las que su funcionamiento es correcto. Por ello, es necesario seleccionar de entre los sistemas de predicción aquellos que proporcionen pronósticos precisos incluso cuando partan de información con mucha desproporción entre el número de ejemplos que representan los aciertos y el número de ejemplos que representan los fallos. A este tipo de problemas se les denomina *desequilibrados*.

En la literatura de Minería de Datos existen múltiples opciones para desarrollar sistemas de Reconocimiento de Patrones, pudiendo todas ellas alcanzar precisiones razonables en el análisis de ciertos problemas industriales. Sin embargo, las restricciones anteriormente detalladas han de ser tenidas en cuenta a la hora de seleccionar el sistema de predicción más adecuado. En este contexto, cabe destacar que el planteamiento que se busca en el presente estudio es diferente del de otros trabajos anteriores que utilizan la Minería de Datos para analizar los problemas industriales considerados. Así, frente a otros trabajos

en los que se analiza únicamente un sistema de predicción, en este se busca comparar las prestaciones de diversos métodos, seleccionado aquellos que más se ajusten a las limitaciones presentes en los entornos de producción.

Por otra parte, hay algunas técnicas de Reconocimiento de Patrones que son muy populares en el área de Minería de Datos, pero se han empleado con menos frecuencia en el ámbito de aplicaciones industriales y que se pretenden incluir en el presente estudio. Tal es el caso de los *ensembles*, cuyas altas prestaciones han sido probadas en diferentes ámbitos. Para estas técnicas, existe además un marco teórico que justifica el aumento de precisión que proporcionan en base a la reducción de las componentes sesgo y varianza del error. Dicho análisis se considera fundamental dada la limitación en el tamaño de los datos, lo que hace necesario obtener el máximo de información a partir de pocos ejemplos.

Es importante indicar que este trabajo ha sido posible gracias a la colaboración de tres grupos de investigación, que han proporcionado los datos usados en la experimentación. Por un lado, para el diagnóstico de fallos en aerogeneradores, el área de Diagnóstico Industrial y Mantenimiento Predictivo de la Fundación CARTIF ha proporcionado un conjunto de datos generado en un banco de pruebas a escala que simula el funcionamiento de un aerogenerador. Por otro lado, el análisis de procesos de fabricación de piezas metálicas de geometría compleja ha sido posible gracias a dos conjuntos de datos suministrados por dos universidades que disponen de equipos de fabricación que utilizan tecnología láser. Así, el Departamento de Ingeniería Mecánica y de Construcción Industrial de la Universitat de Girona recogió datos sobre micro-fresado 3D sobre acero inoxidable 316L utilizando un láser de Nd:YAG a 1064 nm de longitud de onda, mientras que el Departamento de Ingeniería Mecánica de la Universidad del País Vasco realizó pruebas sobre el pulido superficial de dos materiales, una aleación comercial LaserForm^{TM}ST-100 y un acero Orvar Supreme, mediante láser de CO_2.

1.1. Objetivos

El presente trabajo tiene como objetivo general aumentar la eficiencia de dos procesos industriales: el diagnóstico de fallos en aerogeneradores y la fabricación de piezas metálicas de geometría compleja, utilizando técnicas de Minería de Datos. Para ello, se pretende mejorar el procedimiento de validación experimental de estudios anteriores, en los que no

se ha usado validación cruzada estratificada ni se han tenido en cuenta algunas particularidades de los problemas analizados, como el desequilibrio entre fallos y aciertos, para el diagnóstico de fallos en aerogeneradores, y la cantidad limitada de datos, como ocurre en el caso de fabricación de piezas metálicas de geometría compleja.

Este objetivo general se desglosa en los siguientes objetivos específicos:

1. Diagnóstico automático de fallos en aerogeneradores.
    - Identificación, por validación cruzada, de la técnica de clasificación más adecuada para relacionar las medidas de vibraciones con el tipo de fallo.
    - Estudio de la influencia del desequilibrio en los datos en la precisión de los clasificadores.
    - Definición de la métrica más adecuada para evaluar la precisión de los clasificadores, considerando el desequilibrio existente en los datos.
2. Estimación de la calidad superficial en el pulido láser.
    - Como paso previo, reescritura de todos los datos en un formato adecuado para las técnicas de clasificación, teniendo en cuenta el estándar industrial que define la calidad superficial.
    - Identificación, por validación cruzada, de la técnica de clasificación más adecuada para predecir la calidad superficial.
3. Estimación de la calidad en la obtención de geometrías complejas mediante microfresado 3D láser.
    - Identificación, por validación cruzada, de la técnica de regresión más adecuada para predecir los errores en los distintos requerimientos geométricos.

1.2. Estructura de la tesis

Este documento está organizado en seis capítulos, correspondiendo este capítulo con la introducción. En el Capítulo 2 se introducen los procesos industriales tratados. Por un lado,

se explican los fundamentos físicos de los procesos y su uso actual a nivel industrial. Por otro, se enumeran los trabajos previos del Estado del Arte que analizan dichos procesos, subrayando las aportaciones del presente trabajo respecto de los mismos.

Aparte del Capítulo 5, que contiene las publicaciones, el más amplio de esta memoria es el Capítulo 3. En él se introduce de forma general la Minería de Datos, disciplina dentro de la que se encuadran las técnicas usadas para analizar los procesos industriales. Un punto clave para entender la novedad que presenta este estudio frente a trabajos anteriores es la validación experimental de los predictores, ya que con la realización de validaciones cruzadas estratificadas se consigue mayor rigor en la validación experimental. La parte central del capítulo recoge la descripción de las distintas técnicas de predicción usadas. No se ha incidido con el mismo detalle en todos los tipos de técnicas, sino que se ha prestado más atención sobre aquellas que han proporcionado mejores resultados para el análisis de los problemas industriales considerados. Por ello, en este capítulo se profundiza en la descripción de los ensembles analizados en el estudio junto con las técnicas inductivas en las que éstos se basan. Para finalizar el capítulo, se explican algunas particularidades de dos tipos de problemas de Minería de Datos que han sido analizados en el presente Trabajo de Investigación: la Clasificación Ordinal y la Clasificación Desequilibrada.

En el Capítulo 4 se explican los fundamentos teóricos que justifican la mayor precisión de los *ensembles* en ciertas tareas de predicción. Para ello, se hace especial hincapié en los conceptos de *sesgo* y *varianza*, aunque también se explica el concepto de *margen*, en el que se basan otros autores. En primer lugar se introducen de forma intuitiva las ideas de sesgo y varianza, utilizadas en Estadística Inferencial al referirse a la Estimación de Parámetros de una variable aleatoria. Posteriormente, se analiza cómo pueden extenderse dichos conceptos a problemas de regresión y a problemas de clasificación. En este último caso no existe un consenso en la literatura de Minería de Datos, sino que conviven varias definiciones propuestas por varios autores. Por ello, se realiza una comparativa de varias formulaciones, explicando cuáles son las definiciones que se ha creído más conveniente tomar en cuenta en el estudio.

En el Capítulo 5 se exponen las publicaciones que forman parte de esta Tesis Doctoral. En total, este trabajo contiene cinco publicaciones. Por un lado, se han presentado dos publicaciones referentes a la fabricación mediante tecnologías láser: "*Improvements in modelling of complex manufacturing processes using classification techniques*" y "*Modelling*

laser milling of micro-cavities for the manufacturing of DES with ensembles". Por otro, se han presentado tres estudios sobre el mantenimiento de aerogeneradores: "*Wind turbines fault diagnosis using ensemble classifiers*", "*An SVM-Based Solution for Fault Detection in Wind Turbines*" y "*Identifying maximum imbalance in datasets for fault diagnosis of gearboxes*". Finalmente, en el Capítulo 6 se enumeran las principales conclusiones de este trabajo y las futuras líneas de actuación.

Capítulo 2

Procesos industriales considerados

Los dos problemas industriales analizados en el presente estudio son la fabricación de piezas metálicas de geometría compleja mediante tecnologías láser (Sección 2.1) y el diagnóstico de fallos en aerogeneradores (Sección 2.2). En ambos casos se ha dividido la exposición en dos apartados: el Contexto Industrial y el Estado del Arte.

En el primer apartado se explican los fundamentos físicos de los procesos industriales estudiados, contextualizando además la situación actual de estos procesos a nivel industrial, considerando la evolución que se ha producido en los últimos años y las demandas futuras. En el segundo apartado se enumeran los trabajos previos que también han analizado estos procesos industriales y se indican cuáles son las aportaciones y mejoras respecto de los mismos.

2.1. Fabricación de piezas metálicas complejas

2.1.1. Contexto Industrial

A lo largo de la dos últimas décadas se ha producido una mejora sustancial en la capacidad tecnológica de fabricación de piezas metálicas, gracias, en muchos casos, a la aparición de nuevas tecnologías de fabricación, como los sistemas de fabricación por adición láser entre los que se encuentran la *Fusión Selectiva por Láser* (*Selective Laser Melting*, SLM, en la bibliografía inglesa) o la *Fusión por Haz de Electrones* (*Electron Beam Melting*, EBM, en la bibliografía inglesa). Esto ha permitido producir piezas de geometría y

complejidad anteriormente imposibles con otras técnicas tradicionales de mecanizado y, además, con unos tiempos de fabricación más reducidos [167]. Pero para estos procesos aún están por determinar las condiciones de operación óptimas, a diferencia de lo que sucede con otras tecnologías de fabricación más tradicionales. Esta carencia viene dada por el hecho de que la naturaleza que rige estas nuevas formas de fabricación le es ajena a los ingenieros de proceso, ya que son tecnologías de reciente desarrollo en los que se tiene una experiencia muy limitada a nivel industrial.

Existen dos aspectos diferentes a la hora de valorar la calidad final de una pieza metálica. Por un lado, la calidad de la geometría obtenida (en qué grado son coincidentes las proporciones de la pieza obtenida a las programadas inicialmente), y, por otro, la calidad del acabado superficial. Por ello, en la producción industrial de piezas metálicas se distingue entre una etapa dedicada a obtener la geometría y otra fase posterior centrada en el acabado superficial. Por un lado, en lo referente a la calidad en la obtención de la geometría de una pieza, hay que tener en cuenta que las dimensiones de una pieza no se expresan de forma exacta, sino que se indican a través de un intervalo, conocido como *tolerancia* [84]. Por otro lado, en el caso de evaluar la calidad superficial de una pieza, ésta no es perfectamente regular, sino que presenta irregularidades. Sobre dichas ondulaciones se mide la *rugosidad*, que se calcula a partir de la diferencia de altura entre los picos y los valles [202]. Mientras que las tolerancias de todas las dimensiones lineales y angulares de cada pieza se definen a través de los estándares ISO 286 y ISO 2768, es el estándar ISO 4288 el usado para medir la calidad superficial.

En el presente trabajo se ha abordado el estudio de la calidad de fabricación de piezas metálicas de geometría compleja, tanto en la etapa de obtención de la geometría como en la etapa en la que se consigue el acabado superficial necesario. Para ello, se han analizado dos procesos que usan las tecnologías de fabricación mediante láser: el micro-mecanizado 3D y el pulido superficial. El láser es un dispositivo que genera luz con alto contenido energético dado que la luz láser es coherente, monocromática y direccional [85]. Estas propiedades permiten su aprovechamiento industrial, concentrando su energía en los puntos concretos de la superficie de la pieza a tratar.

Los dos procesos estudiados no tienen que usarse necesariamente en la fabricación de un mismo producto, sino que han sido elegidos porque son novedosos e implican la interacción del láser con la materia, un fenómeno físico de gran complejidad en el que

resulta especialmente útil la búsqueda de modelos predictivos a partir de las variables de control del proceso. Las dos técnicas de fabricación analizadas son:

- **Micro-mecanizado 3D** por láser (en la bibliografía inglesa, *laser micro-machining*, o también ablación láser, *laser ablation*), que se encuadra dentro de la etapa de obtención de la geometría. La ablación láser es un proceso por el cual se elimina material de una pieza mediante un haz láser [161]. Como se ve en la Figura 2.1, al incidir el haz de luz láser sobre la pieza a tratar, se desprende de la misma parte de materia, que sale de la superficie en forma de un chorro de plasma. Específicamente, dentro de las técnicas de micro-mecanizado, se analiza el **micro-fresado** (*micro-milling*). Este estudio se recoge en la publicación "*Modelling laser milling of micro-cavities for the manufacturing of DES with ensembles*" (sección 5.1). La diferencia entre un fresado en 2D y otro en 3D reside en la posición que la herramienta de corte con la fresa puede tomar respecto de la pieza. En un fresado 2D la herramienta se posiciona en un ángulo constante respecto de la superficie de la pieza, avanzando en 2 ejes, mientras que un tercer eje controla la profundidad del corte. Sin embargo, en un fresado 3D la herramienta puede orientarse hasta tener el ángulo deseado con la superficie a tratar, siendo posible eliminar material de forma continua con cualquier inclinación. De este modo, mientras que con el fresado 3D se obtienen geometrías muy complejas (como las demandadas por el sector aeronáutico), el fresado 2D suele utilizarse para producir piezas con formas prismáticas o de geometría más sencilla (por ejemplo, las empleadas en la producción de moldes y matrices).

- **Pulido superficial** láser (en la bibliografía inglesa, *laser polishing*, o también refundición del láser, *laser remelting*), que se encuadra dentro de la etapa de mejora del acabado superficial. En el pulido láser, se funde una fina capa del material con una tensión superficial que hace que el material fluya desde los picos hasta los valles (ver Figura 2.2). No hay por tanto eliminación de material, sino un reposicionamiento del mismo al fundirse. Para este segundo caso, el análisis se corresponde con la publicación "*Improvements in modelling of complex manufacturing processes using classification techniques*" (sección 5.1).

Existen modelos físicos que determinan el resultado del tratamiento de piezas metálicas con los dos procesos estudiados. Sin embargo, en ambos casos los modelos dependen

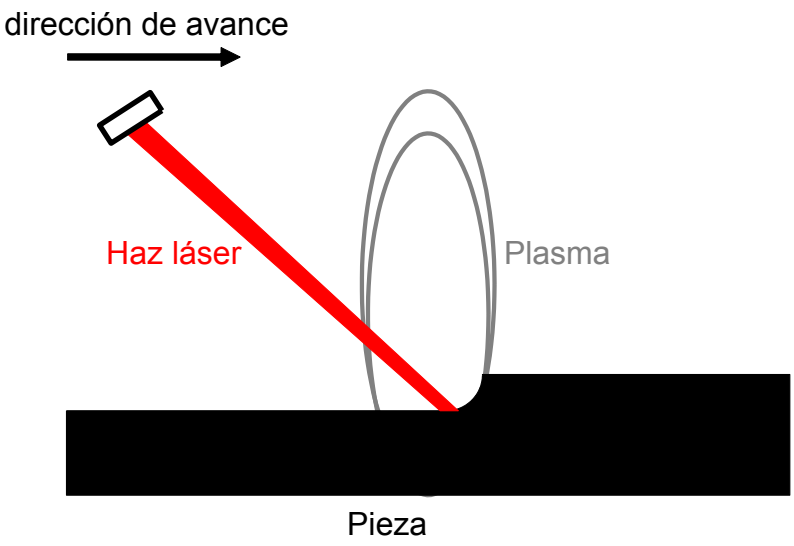


Figura 2.1: Ablación láser

Figura 2.2: Pulido láser

de la densidad de energía (energía por unidad de área), que no puede ser determinada con exactitud en la práctica a nivel industrial, ya que es función del área del haz de luz que incide sobre la superficie [39]. Mientras que para el mecanizado la densidad de energía es el único factor que se considera determinante, [143], para el pulido el resultado del proceso depende también de la rugosidad de la superficie a tratar [39].

Aunque las características de la superficie a tratar sí pueden ser determinadas con un grado de error pequeño [151], la densidad de energía sólo se conoce de forma teórica, tanto para el mecanizado como para el pulido, puesto que depende del área del haz de luz incidente y de la forma de la distribución de la energía en el haz. En teoría, el área y la distribución del haz dependen del tipo de láser usado y de sus propiedades ópticas, pero en realidad varía de forma sustancial con la distancia entre el foco y el punto de interacción entre el láser y la superficie del material y el ángulo de incidencia del haz [78] y, en menor medida, con el ajuste del sistema láser y su antigüedad. En un entorno de producción es especialmente difícil de controlar el ángulo de incidencia del haz, puesto que está sometido a variaciones debido a rugosidades superficiales de la pieza y no siempre se dispone de medios para caracterizarlo.

En la práctica, los parámetros que se controlan a nivel industrial son la velocidad de avance del haz, su frecuencia y el porcentaje de intensidad respecto del máximo posible. En el presente trabajo se han usado técnicas de Minería de Datos para predecir el resultado de los procesos analizados en función de estas tres variables que sí son controlables industrialmente.

2.1.2. Estado del Arte

Existen diversos estudios en lo referente a la optimización la calidad de las piezas obtenidas mediante tecnologías láser. Todos ellos tratan de determinar una relación entre los parámetros de control del proceso (como la velocidad de avance del láser o el gas auxiliar) y una variable de salida que evalúa la calidad de la pieza, pero siguiendo tres enfoques diferenciados [62]:

1. Estudios experimentales - Describen de forma cualitativa el efecto de variar parámetros de control del proceso sobre el rendimiento del proceso, sin establecer una relación matemática explícita entre los parámetros de control y la calidad final.

2. Estudios analíticos - Establecen de forma explícita la relación matemática entre las entradas del proceso y una medida de su rendimiento. Por tanto, permiten analizar la influencia individual de cada parámetro en el resultado final, buscando un mayor entendimiento del proceso.

3. Modelos de inteligencia artificial - El objetivo es encontrar una relación lo más precisa posible, aunque el modelo no permita expresar de forma explícita una relación entendible entre los parámetros de control del proceso y la calidad de la pieza.

Si nos centramos en el **mecanizado láser**, dentro de la primera categoría se tienen numerosos ejemplos para el análisis de mecanizado a escala macroscópica. Así, diversos autores han analizado la influencia de la velocidad de barrido y la intensidad y frecuencia del láser en la calidad final de las piezas sobre diferentes materiales [42, 50, 52, 163]. También hay diversos trabajos de análisis de mecanizado a escala microscópica, tales como el micro-taladrado (*micro-drilling*) [6, 21, 206], el micro-corte (*micro-cutting*) [127, 136, 187] y el micro-fresado en 2D (*2D micro-milling*) [33, 49], pero hay poca documentación en lo referente al micro-fresado 3D (*3D micro-milling*).

En lo referente a los modelos analíticos, Karanakis et al. [102] probaron la capacidad de un láser de picosegundos de fresar diferentes materiales (acero inoxidable, óxido de aluminio -alúmina- y dióxido de silicio fundido -sílice-). Se correlacionó información sobre la topología de la superficie con la densidad de la energía incidente, para identificar el óptimo en el procesamiento. Qi et al. [147] usaron un láser de fibra para mecanizar geometrías complejas. Desarrollaron un modelo térmico de la ablación para determinar el volumen de material eliminado y sus dimensiones, así como optimizar los parámetros para alcanzar la máxima eficiencia con los mínimos efectos térmicos. Teixidor et al. [184] estudiaron los efectos que tenían la velocidad de barrido, la intensidad del pulso y su frecuencia sobre la anchura, profundidad y rugosidad de la superficie, para el caso de fresado con láser de micro-canales en acero para mecanizado. Presentaron un modelo de segundo orden y un proceso de optimización multi-objetivo para predecir las respuestas y la combinación óptima de parámetros. Finalmente Vázquez et al. [193] analizaron el micro-fresado de cavidades en moldes, incluyendo aspectos como la geometría de la herramienta de corte, la segmentación de la geometría de la cavidad, el espesor de viruta, la tolerancia cordal y el tipo de interpolación.

Por último, se citan los estudios de la tercera categoría, que utilizan técnicas de Inteligencia Artificial. Muchos de ellos se refieren a la escala macroscópica [44, 51, 186], pero hay pocos ejemplos para el micro-fresado láser [62]. Tres trabajos usaron redes neuronales para elaborar el sistema de predicción: en [209], para predecir la energía del pulso necesaria para obtener la profundidad y el diámetro deseado mediante el micro-fresado; en [42], la tasa de eliminación de material para una profundidad de ablación fija dependiendo de la velocidad de barrido y la frecuencia del pulso, y en [96], la calidad de procesos de corte de chapas de 1 mm de espesor en aceros de construcción. Finalmente, se propusieron árboles de regresión para optimizar las dimensiones geométricas en la fabricación de micro-canales [185].

En lo que se respecta al **pulido superficial láser**, apenas han sido usadas técnicas de Inteligencia Artificial para optimizar los procesos de fabricación de piezas metálicas. Así, en una revisión del año 2013 de los distintos análisis para evaluar la calidad superficial obtenida con el pulido superficial láser, Bordatchev Hafiz y Tutunea-Fatan distinguen dos tipos de estudios [24]:

- Análisis experimental. Se varían varios parámetros de de entrada del proceso en un rango, midiendo la influencia que tienen varios sobre el acabado superficial o el tiempo de mecanizado. Algunos de los parámetros que se han considerado han sido la fluencia y número de pulsos incidentes [16], la potencia y el desenfoque [180], la tasa de alimentación, frecuencia y energía del láser y duración del pulso [76, 89, 90], el tipo de distribución energética y forma del haz de luz láser [139], y algunos parámetros de control del haz de luz láser y la topografía inicial de la superficie [115].

- Estudios de modelado y simulación. Tratan de establecer una relación analítica entre la calidad de las piezas obtenida y varias variables del proceso, a través de modelos teóricos. Algunos de ellos se validaron mediante simulaciones numéricas en base al método de elementos finitos, mientras que en otros no existió tal validación. Así, se han propuesto modelos en los que se asimilan las asperezas superficiales con capas hemisféricas [153], modelos termo-físicos [152] y modelos que analizan el pulido láser como un cambio de estado sólido/líquido, derivados del modelo clásicos que analiza las fronteras entre cambios de fase, conocido como problema de Stefan [122].

Aunque para otros procesos industriales láser en los cuales se derrite el material como la soldadura y el SLM sí se han realizado numerosos análisis basados en técnicas de Minería de Datos [7, 23, 53, 105, 105, 113, 175], no hemos encontrado ningún trabajo en la literatura que estudie el pulido superficial láser con técnicas de Minería de Datos, aparte de un trabajo previo sobre el mismo conjunto de datos, pero empleando regresores en vez de clasificadores [39].

2.1.3. Aportaciones respecto del estado del arte

Aunque existen otros estudios basados en Inteligencia Artificial que analizan los dos procesos considerados, dichos estudios presentan las siguientes limitaciones:

- Para el micro-fresado 3D, las técnicas de predicción usadas han sido muy limitadas, centrándose en las redes neuronales. En el presente trabajo, se han utilizado ensembles, analizando de forma teórica y expermiental la causa de la disminución de error de predicción que proporcionan en base a las componentes sesgo-varianza. En primer lugar, cabe resaltar que este tipo de técnicas han obtenido buenos resultados experimentales en muchas aplicaciones industriales [38, 141]. Además, se han aplicado para modelos de procesos de mecanizado, si bien a escala macroscópica, como el fresado y el taladrado [37, 40, 60, 65], donde resultan especialmente útiles porque pueden alcanzar gran precisión con menor tiempo de sintonización de los parámetros del modelo [37].

- Para el pulido superficial, sólo se encontró un trabajo anterior que analizaba el problema con técnicas de Minería de Datos, pero mediante técnicas de regresión. En el presente estudio se han empleado técnicas de clasificación ordinal, obteniendo mayor precisión. Además, no se tuvieron en cuenta todos los parámetros del proceso que tienen influencia a nivel industrial, excluyéndose aspectos como el tipo de gas auxiliar utilizado o el ángulo de incidencia del haz, aspectos que sí se han considerado en el presente trabajo.

2.2. Diagnóstico de fallos en aerogeneradores

2.2.1. Contexto Industrial

Desde comienzos del presente siglo ha habido un aumento progresivo en la producción de energía eléctrica a través de aerogeneradores, debido tanto a la creciente demanda de energía como a la búsqueda de fuentes de energía menos contaminantes que las basadas en el uso de combustibles fósiles. Sin embargo, diversos estudios realizados en los últimos años muestran que el coste de producción con este tipo de centrales es superior al de otros sistemas de producción de energía. Así, un estudio comparativo de precios de producción de energía eléctrica del año 2008, sitúa los costes de producción con gas natural en 4,9 céntimos de euro el kWh, 4,1 para el carbón y 6,6 para la nuclear [129], frente a los 6,6 de los parques eólicos terrestres y los 8,6 de los parques eólicos marinos [22].

El interés en abaratar los costes de producción a través de parques eólicos hasta equipararlo al de otras fuentes de generación más tradicionales ha conllevado el desarrollo de una serie de estudios orientados a mejorar su mantenimiento, considerado el elemento más encarecedor de este tipo de producción [99]. Las técnicas clásicas de mantenimiento de máquinas rotativas constituyen el punto de partida para estos nuevos estudios. Sin embargo, es necesario tener en cuenta que los aerogeneradores trabajan en régimen de velocidad variable, algo que los diferencia de otras máquinas rotativas y que hace necesario un enfoque específico del mantenimiento de estos dispositivos [176].

El diagnóstico clásico del estado de maquinaria rotativa se centra en el nivel de vibraciones. Esto es debido a que en el funcionamiento de una máquina no es posible aprovechar toda la energía en trabajo útil, sino que cierta cantidad de energía se disipa en forma de vibraciones, a causa de la transmisión de fuerzas cíclicas entre distintos mecanismos [135]. Aunque es posible reducir la vibración a niveles bajos con un diseño adecuado, el funcionamiento de los equipos hace que éstos se vayan desgastando con el tiempo, apareciendo un deterioro de la respuesta mecánica, y por consiguiente un aumento de las vibraciones.

El método más común de análisis de fallos en máquinas rotativas es el análisis de vibraciones en el espacio transformado de Fourier [56]. Como se muestra en la Figura 2.3, el espectro de Fourier representa el nivel de vibración registrado para cada frecuencia. En dicha figura se han representado dos señales (parte superior) y su correspondiente transfor-

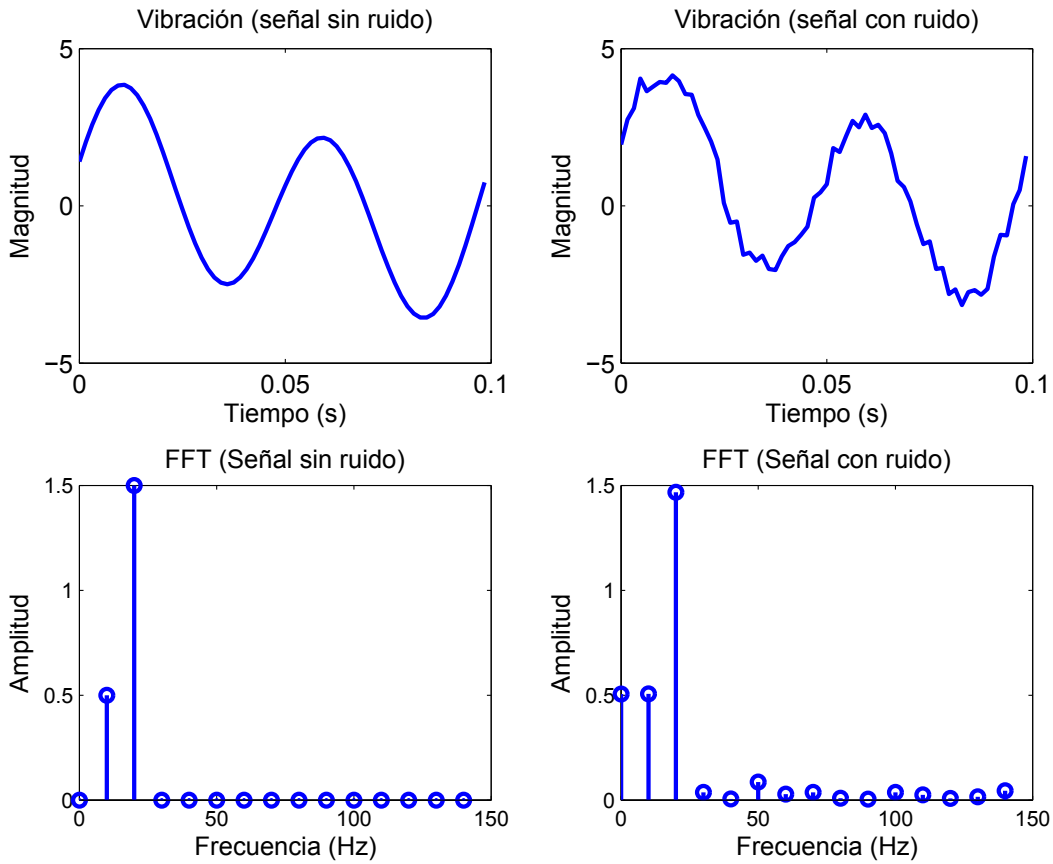


Figura 2.3: Análisis de vibraciones con la Transformada rápida de Fourier (FFT)

mada (parte inferior). En la parte izquierda está una onda compuesta por dos vibraciones, con frecuencias de 10 y 20 Hz, y en la parte derecha se ha añadido un ruido aleatorio a la señal. Se observa como el ruido no modifica significativamente las dos componentes en las frecuencias de 10 y 20Hz de la transformada, sino que simplemente queda reflejado en la componente continua (barra a 0Hz en la figura), sufriendo el resto de componentes solamente ligeras variaciones.

El método estándar de cálculo de las componentes en el espacio transformado es la Transformada Rápida de Fourier (*Fast Fourier Trasform, FFT*, en la bibliografía inglesa). Sin embargo, cuando se analiza una señal de velocidad variable, no se obtiene directamente por dicho método de cálculo información útil para el análisis de vibraciones. Esto es debido a un problema conocido como "manchado espectral" (*smearing effect*, en la bibliografía inglesa), que si bien se presenta siempre en la FFT cuando la señal analizada

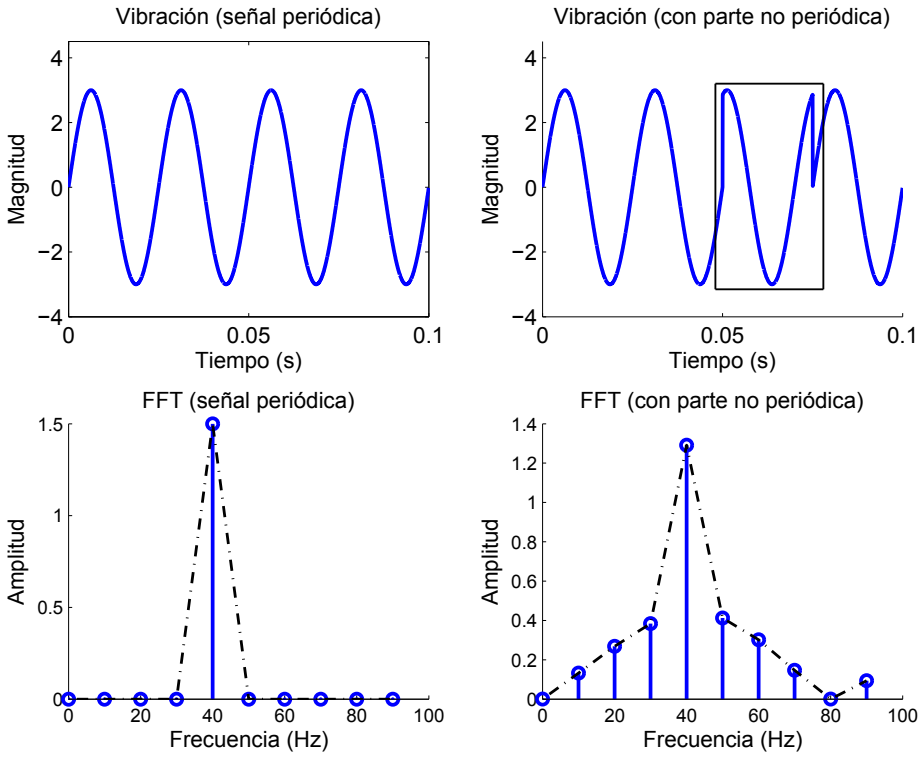


Figura 2.4: Manchado espectral de la FFT

no es totalmente periódica, se ve agravado con la variabilidad de velocidad. La no periodicidad de la señal provoca en la FFT que se atenúe la amplitud en las componentes principales de frecuencia, traspasándose parte de su valor a las frecuencias adyacentes. Esta aparición de componentes en frecuencias donde realmente no hay vibración se conoce como "fuga espectral" (*spectral leakage*, en la bibliografía inglesa). En la Figura 2.4 se ilustra este fenómeno tomando como referencia una señal con una única vibración a 40Hz de frecuencia. En la parte izquierda se muestra la señal totalmente periódica y su correspondiente FFT, mientras que en la derecha se tiene una señal que no es totalmente periódica, ya que en la parte de la señal marcada con un recuadro negro se ha variado la fase.

El seguimiento de órdenes (*order tracking*, en la bibliografía inglesa) es una técnica de procesamiento de la señal que permite obtener un espectro de Fourier tomando como referencia la velocidad de giro de la máquina en cada momento, en lugar de tomar valores estáticos de referencia. De esta forma, mientras que un espectro de Fourier directo tiene

como variable en el eje horizontal la velocidad de giro en hertzios, en un espectro obtenido con seguimiento de órdenes, lo que se tiene en el eje horizontal son órdenes (múltiplos de la velocidad de giro que tiene en cada momento la máquina). Para ello, a partir de la señal medida, muestreada a intervalos constantes de tiempo, debe obtenerse la transformada de Fourier de otra señal, muestreada a intervalos constantes del ángulo del eje de rotación.

Cabe destacar que para obtener este espectro de Fourier de la señal transformada no tiene porque ser necesario calcular explícitamente dicha señal transformada. Así, dentro del análisis de órdenes se distinguen tres familias de métodos:

- "Seguimiento de órdenes computado" (*Computed order tracking*), si la señal transformada se calcula explícitamente [70]. En este caso al referirse a la transformación de la señal de dice que se *pasa del dominio temporal al dominio angular*.

- "Transformadas de Seguimiento de Órdenes" [25], si se estima cuál es la formulación completa del espectro que tiene la señal transformada, pero sin calcularla explícitamente.

- El filtro de Vold-Kalman [119] calcula el espectro a partir de estimaciones por interpolación de una serie de pares de valores discretos (órdenes, amplitud), sin necesidad de usar la señal transformada.

En el presente trabajo, el análisis de vibraciones se encuadra dentro de la primera categoría, al calcularse explícitamente la señal en el dominio angular. En concreto se utilizarán unos datos de vibración que han sido tratados previamente mediante el *remuestreo angular* [194], una mejora del método propuesto por Fyfe y Munck [70]. En la Figura 2.5 se muestra la señal inicial a la izquierda de la imagen, muestreada a intervalos fijos de tiempo, y la remuestreada a la derecha, a intervalos fijos de ángulo.

En el presente estudio se han analizado los dos fallos más usuales en los aerogeneradores, el desalineamiento, cuando el eje de la turbina no es paralelo al resto del eje de transmisión de la máquina, y el desequilibrio, cuando el centro de gravedad de la turbina no coincide con su centro de rotación [146]. En las Figuras 2.6 vemos una representación de ambos tipos de fallo.

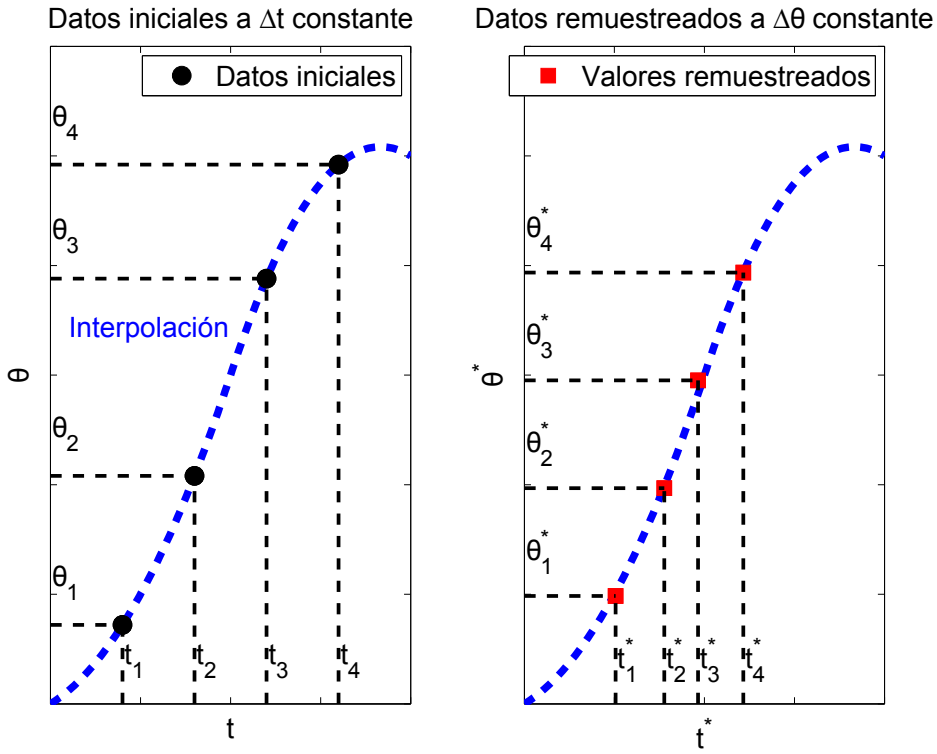


Figura 2.5: RemuestreoAngular

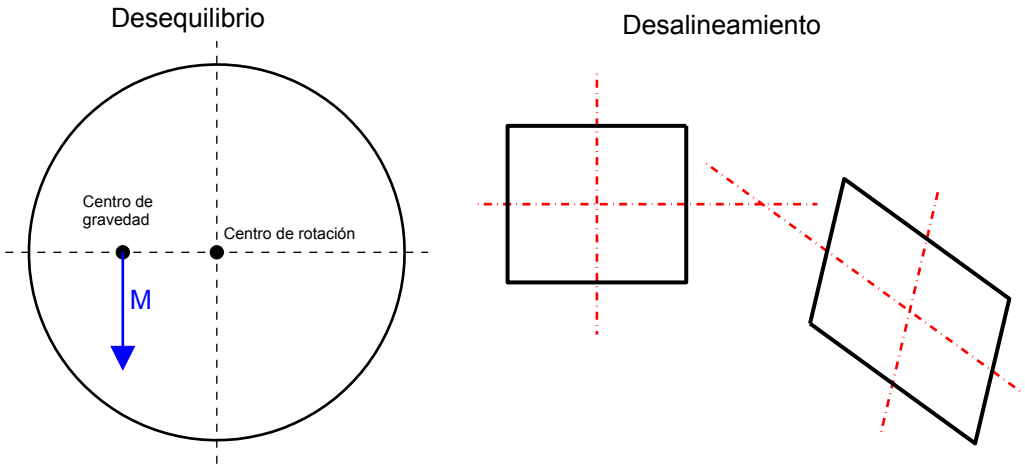


Figura 2.6: Desequilibrio y Desalineamiento

2.2.2. Estado del Arte

Como se ha explicado en el apartado anterior, la detección de fallos mecánicos en máquinas rotativas se basa en el análisis del nivel de vibración que presentan. Dicho análisis conlleva tres tareas: recoger datos con el nivel de vibración, procesarlos de forma que sean interpretables y diagnosticar el estado de funcionamiento de la máquina. En la literatura sobre diagnóstico de fallos en aerogeneradores se encuentran estudios con dos tipos de objetivos diferenciados. Por un lado, hay trabajos que se centran únicamente en la tarea de *recoger información* que sea útil para identificar el estado de funcionamiento de los aerogeneradores. Pero, por otro lado, hay estudios que además analizan como *elaborar un sistema de decisión* a partir de los datos almacenados. En este segundo enfoque se pueden además distinguir dos alternativas: que la decisión la tome un experto o automatizar el proceso a través de técnicas de Inteligencia Artificial. Por tanto, es posible dividir los trabajos previos en tres categorías:

1. Estudios sobre la definición de atributos que caractericen el estado de funcionamiento de los aerogeneradores. Estos trabajos inciden tanto en los sensores como en el procesamiento de la información de los mismos hasta definir unos atributos útiles.

2. Estudios sobre cómo desarrollar un sistema de toma de decisión en base al criterio de un experto.

3. Estudios sobre cómo desarrollar un sistema de toma de decisión automático mediante técnicas de Inteligencia Artificial.

Dentro de la primera categoría, inicialmente se consideró la posibilidad de estudiar las vibraciones directamente en el dominio temporal [165], mediante análisis estadísticos de las señales de vibración. Posteriormente, se demostró que caracterizar la señal mediante un análisis espectral o en el dominio tiempo-frecuencia era más eficaz a la hora de extraer la mayor cantidad de información posible [10, 179, 212]. Recientemente, los análisis espectrales han sido complementados con la técnica del remuestreo angular, para compensar el efecto de la variabilidad del viento [194, 195].

Una vez que la información se ha recogido mediante sensores y procesado para transformarla en información analizable, aún es necesario desarrollar un sistema de decisión

para el diagnóstico de fallos. Entre los trabajos que basan la toma de decisión en un experto podemos destacar el método de la *kurtosis espectral* [9], el *análisis espectral singular* [166], la *descomposición modal empírica* con *análisis independiente de componentes* [197] y el uso de modelos de regresión [194, 195, 212].

Finalmente, en la tercera categoría, se incluyen aquellas técnicas basadas en la Inteligencia Artificial, mediante las cuales no es necesaria la presencia de un experto para identificar cuáles son las variables críticas [101]. Así, diversos autores han utilizado técnicas de Minería de Datos para predecir fallos en aerogeneradores. Para ello, hay estudios que toman como atributos datos medidos por un SCADA (*Supervisory Control And Data Acquisition*, sistemas de control y adquisición de datos) durante el funcionamiento de los generadores, mientras que en otros trabajos se utilizan bancadas de ensayos para simular fallos y se recoge información del nivel de vibración mediante acelerómetros.

De entre los estudios que utilizan bancos de ensayos con acelerómetros, se han empleado diversas técnicas de clasificación como un árbol binario de Máquinas de Vectores Soporte (SVM) con una estructura pre-seleccionada en base a un mapa auto-organizado SOM [201], redes bayesianas [48], o una combinación de árboles de decisión y sistemas expertos [207]. Wenyi et. al [201] no usan directamente como atributos los coeficientes de la transformada discreta de Fourier, sino el *bi-espectro*, a través del cual analizan si ha habido un fallo por rotura de diente de un engranaje interno o externo. El *bi-espectro* es un tipo de *especto superior*, también llamado *espectro cumulante* o *poliespesctro*, cuyos coeficientes se calculan a partir de los coeficientes de la transformada discreta de Fourier y reciben el nombre de *cumulantes* [133]. Dichos coeficientes están relacionados con momentos estadísticos de orden superior a dos y presentan la ventaja de revelar información respecto a la fase de la señal, información que no aportan los momentos estadísticos de orden dos, como la correlación [41]. Por otra parte, en el trabajo de Chen y Hao [48] se tienen en cuenta únicamente a 15 atributos binarios. Por un lado, hay atributos que son resúmenes estadísticos de la distribución de la señal (como la media, el valor pico, la amplitud o el apuntamiento), y, por otro, atributos referidos a las frecuencias naturales de vibración del sistema. En todos los casos, sólo se consideran dos posibles valores de los atributos: normal o anormal. A partir de dichos atributos proponen una metodología para predecir 6 tipos de fallos: error de perfil de diente, rotura de un diente de los engranajes, flexión del eje, desequilibrio en el eje, fatiga en los rodamientos, resonancia. Yongxin

analizó siete tipos de fallos por roturas de dientes, fatiga superficial (picadura por desconchado) y fallas estructurales tanto en la rueda de mayor como de menor velocidad [207]. Para ello, los clasificadores usaron 13 atributos definidos a partir de un sistema supervisado de extracción de las características de las señales de vibración relacionadas con los fallos [188].

En los tres trabajos anteriormente citados la técnica de validación experimental presentó severas limitaciones. Así, el trabajo de Chen y Hao aporta un ejemplo práctico de cálculo de probabilidad de fallo con su sistema, pero no valida dicho sistema experimentalmente, mientras que en los de Wenyi et al. y Yongxin se usó una parte de los datos para entrenar y otra para realizar el test. No se usó validación cruzada ni se tuvo en cuenta el desequilibrio entre la proporción de fallos y el número de situaciones de funcionamiento correcto.

En lo referente al uso de un SCADA para recoger los atributos, Kusiak y Li propusieron un sistema de diagnóstico de fallos de tipo predictivo [114] , en el que se relacionaban los fallos con valores de variables medidas con una hora de anterioridad (12 muestreos del SCADA a 5 minutos cada muestreo). Se aplicaron cuatro técnicas de minería de datos: redes neuronales, un ensemble de redes neuronales, ensembles de árboles de decisión tipo Boosting, y Máquinas de Vectores Soporte, tomando como atributos de los clasificadores 60 parámetros, relativos a medidas de velocidad e intensidad del viento, conversión en energía eléctrica, análisis de vibraciones y parámetros de temperatura. Propusieron un sistema de diagnóstico en tres fases. En primer lugar se distingue entre situaciones de funcionamiento normal y situaciones de fallo. En una segunda fase, se clasifican los fallos de acuerdo a cuatro niveles de severidad. Finalmente, en una tercera etapa se concreta qué tipo de fallo específico se tiene, pero debido al gran número de tipos de fallos que puede registrar el SCADA, en la experimentación sólo se considera la distinción entre el tipo de fallo más frecuente y el resto de casos. Tampoco en este caso se empleó validación cruzada ni se tuvo en cuenta el desequilibrio entre la proporción de fallos y el número de situaciones de funcionamiento correcto. Inicialmente disponían de un número mucho mayor de situaciones de funcionamiento correcto que de fallos, pero se muestrearon aleatoriamente un número de de instancias representantes del funcionamiento correcto hasta tener conjuntos con similar número de situaciones de funcionamiento correcto y de fallos.

2.2.3. Aportaciones respecto del estado del arte

Respecto de los trabajos anteriores, el presente estudio aporta mejoras en dos aspectos: la técnica de recolección de datos y la validación experimental. En cuanto a la **la validación experimental**, en anteriores estudios no se usaron validaciones cruzadas, mientras que el análisis del presente estudio se basa en utilizar validaciones cruzadas para medir la precisión de los clasificadores, y aplicar test *t* remuestreados corregidos para comparar la precisión entre varios clasificadores. De esta forma se puede analizar si existen diferencias estadísticamente significativas entre la precisión de diferentes clasificadores, algo que no se ha evaluado en trabajos anteriores.

En cuanto a la **técnica de recolección de los datos**, los anteriores trabajos presentan limitaciones en dos aspectos:

- Los atributos que describen los fallos no recogen un volumen de información suficientemente amplio para que los modelos de inteligencia artificial puedan caracterizarlos completamente.

- Los conjuntos de datos usados anteriormente para analizar el diagnóstico de fallos en aerogeneradores construyen un tipo de problema conocido en Minería de Datos como *clasificación equilibrada* (para más detalles, consultar 3.7). Esto quiere decir que el número de instancias que corresponden con un funcionamiento correcto del aerogenerador es similar al número de instancias que recogen situaciones de fallo. Sin embargo, en condiciones reales de explotación necesariamente se tienen muchos menos fallos que situaciones de funcionamiento correcto (el problema es necesariamente una *clasificación desequilibrada*).

Se distinguen dos enfoques en los trabajos realizados en los últimos años en el campo del diagnóstico de fallos, ambos con limitaciones severas. El primer planteamiento toma acelerómetros como sensores y unas tasas de adquisición de datos a alta frecuencia (del orden de kilohertzios). En este caso, sin embargo, los periodos de tiempo en los que se registran datos (del orden de meses) no son suficientes para tener caracterizadas las situaciones en las que se producen los fallos de forma precisa. Además, ninguno de estos estudios cubre todo el rango de condiciones de un aerogenerador con los regímenes reales de viento, lo que necesariamente conlleva condiciones de funcionamiento con diferente velocidad y carga.

El segundo planteamiento se basa en tomar los sensores de los SCADA, dado que en las condiciones reales de explotación están integrados en el sistema de control de los aerogeneradores; pero estos sistemas de adquisición no tienen acelerómetros, porque su frecuencia de adquisición de datos es baja (del orden de hertzios) y por tanto sus capacidades de predicción son muy limitadas.

La solución es usar bancos de ensayos con acelerómetros, dado que así se recoge información útil y se pueden caracterizar muchos estados de fallo con distintas condiciones de trabajo en un intervalo de tiempo limitado. Pero este sistema de recolección de datos tiene el inconveniente de que el número de instancias que representan cada fallo es similar al número de instancias que representan el funcionamiento normal de un aerogenerador, situación alejada de la que realmente se va a tener al probar en condiciones reales de funcionamiento el modelo de inteligencia artificial entrenado, donde el número de fallos registrados va a ser minoritario.

Esta situación es incluso peor en comparación con otro tipo de máquinas rotativas, debido a la variabilidad de condiciones de carga y velocidad de los aerogeneradores. Mientras que un fallo de una máquina convencional puede ser definido con un número pequeño de instancias, porque sus condiciones de funcionamiento son muy limitadas, la completa identificación de un fallo en un aerogenerador requiere registrar las condiciones de operación del aerogenerador en un rango extenso de condiciones de viento.

En aras de una implementación práctica de los modelos de inteligencia artificial, el análisis realizado en el presente estudio **responde a tres preguntas en lo referente a cómo afecta la presencia de desequilibrio** en los datos:

- ¿Cuál es el máximo nivel de desequilibrio que un determinado modelo de inteligencia artificial puede tolerar, antes de que se produzca una excesiva perdida de precisión en su diagnóstico?
- ¿Cuál es la métrica más adecuada para medir la precisión de un clasificador?
- ¿Qué técnicas de clasificación son más adecuadas para conjuntos desequilibrados multiclase?

La respuesta a la primera pregunta permitirá al ingeniero encargado de las tareas de mantenimiento detectar la técnica más adecuada para el diagnóstico a través del sistema

propuesto. Este proceso de selección dependerá de la disponibilidad de datos adquiridos bajo condiciones de fallo, un parámetro que está directamente ligado a la edad del aerogenerador y su equipo de monitorización.

En lo referente a la segunda cuestión, la métrica más adecuada, hay que tener en cuenta que el porcentaje de instancias correctamente clasificadas es una métrica estándar para conjuntos de datos equilibrados. Sin embargo, no hay un consenso sobre la métrica o conjunto de métricas más adecuadas para los problemas de clasificación desequilibrada multiclase. Se han analizado los valores más comúnmente usados en el estado del arte para clasificación desequilibrada multiclase, escogiendo el más adecuado para este problema industrial.

Finalmente, para responder a la tercera pregunta hay que tener en cuenta que en los trabajos más recientes se han considerado diferentes enfoques, como el remuestreo o aprendizaje sensible al coste. Por ello, se ha realizado una extensa experimentación de diferentes técnicas de clasificación de entre todos los enfoques del estado del arte.

Capítulo 3

Minería de Datos

El análisis de procesos industriales realizado en el presente trabajo se ha llevado a cabo a través de una serie de técnicas de predicción, herramientas proporcionadas por la disciplina conocida como Minería de Datos. En este capítulo se introduce en primer lugar dicha disciplina de forma general, pasando posteriormente a detallar las técnicas de predicción usadas. Las Secciones 3.1 y 3.2 encuadran las herramientas de análisis utilizadas, describiendo en 3.1 los campos de estudio de la Minería de Datos y especificando en 3.2 qué problemas dentro de los anteriores son los considerados en el presente trabajo.

En 3.3 y 3.4 se repasan varios conceptos que es conveniente entender antes de pasar al detalle de la metodología de análisis seguida en el presente estudio. Estos conceptos son fundamentalmente estadísticos y sirven de base teórica tanto a los sistemas de predicción como a su validación experimental. Así, mientras que en 3.3 se describen la Estadística Inferencial y los Contrastes de Hipótesis, en 3.4 se explica la Validación de los clasificadores.

La parte central del capítulo la constituye la Sección 3.5, donde se explican las técnicas de predicción usadas. No se explican con el mismo grado de detalle todos los tipos de predictores, sino que debido a su mayor dificultad o al hecho de haber proporcionado mejores resultados en el presente trabajo hay métodos que se analizan en mayor profundidad. En concreto, se hace mayor énfasis en los ensembles, dado que para los tres procesos industriales estudiados se obtuvieron predicciones más precisas. De hecho, en 4 se incide sobre las causas de los buenos resultados obtenidos por los ensembles.

Finalmente, en las Secciones 3.6 y 3.7 se explican algunas particularidades de dos

tipos de problemas de Minería de Datos tratados en el presente trabajo, la Clasificación Ordinal y la Clasificación Desequilibrada, respectivamente.

3.1. Campos de estudio de la Minería de Datos

El estudio realizado en el presente trabajo se engloba dentro de la disciplina conocida como Minería de Datos (*Data Mining*, en la bibliografía inglesa), encargada de extraer la información relevante de grandes cantidades de datos [80]. Específicamente, se han usado técnicas de Reconocimiento de Patrones (*Pattern Recognition*, en la bibliografía inglesa). Para Fukunaga, el Reconocimiento de Patrones [69] está ligado a la toma automática de decisiones. Así, se busca definir los mecanismos para que un ordenador sea capaz de tomar una decisión de forma automática.

En este contexto, cabe destacar que existen numerosos ejemplos cotidianos de aplicación de sistemas de reconocimiento de patrones a partir de ordenadores, como el reconocimiento de la escritura [118], la voz [91], las imágenes [208] o el correo basura [168].

Dado que ya existe otra disciplina que provee de herramientas para analizar conjuntos de datos, la Estadística, para comprender el área de estudio de la Minería de Datos es conveniente especificar aquellos aspectos que hacen que se haya definido como una disciplina diferenciada de la Estadística [131]. La Minería de Datos surgió de forma natural al ir evolucionando las tecnologías de tratamiento de la información, siendo destacables tres saltos en dicha evolución [95]:

- Recolección de datos y creación de bases de datos.
- Gestión de las bases de datos.
- Análisis de datos y comprensión de las relaciones entre las variables.

Esta evolución del tratamiento de la información fue impulsada desde el sector empresarial, al tratar de obtener, a partir de los datos almacenados en las bases datos, información útil para el proceso de toma de decisión, como parte de lo que se conoce como "Inteligencia Empresarial" (*Business Intelligence*) [18].

Así, la Minería de Datos toma como punto de partida a la Estadística, pero se centra en otros aspectos referidos a la implementación en la práctica de los métodos en un contexto

industrial. Por ejemplo, existen ciertos algoritmos que son paralelizables, lo cual reduce su tiempo ejecución. Además, la Minería de Datos se preocupa de desarrollar técnicas de visualización de datos y define también algunos modelos fácilmente interpretables, tales como los conjuntos de reglas o los árboles de decisión, alejados de un excesivo formalismo usado en ocasiones en los modelos estadísticos [131].

Para Aluja [5], los dos objetivos diferenciados de la Minería de Datos (i.e., los aspectos estadísticos y la necesidad de que los algoritmos sean aplicables en la prácticas) son observables al analizar los orígenes de la Minería de Datos. Así, explica el surgimiento de esta rama de la Ingeniería Informática a partir de la Inteligencia Artificial (en particular el Aprendizaje Automático [*Machine Learning*, en la bibliografía inglesa]) y la Estadística Aplicada (en particular el Análisis Exploratorio de Datos [*Exploratory Data Analysis, EDA*, en la bibliografía inglesa]).

Existe un gran solapamiento entre la Minería de Datos y el Aprendizaje Automático que hace que en ocasiones se confundan dichas disciplinas. Esto es debido a que la Minería de Datos utiliza gran cantidad de técnicas de Aprendizaje Automático, aunque con un objetivo ligeramente diferente [142]. Mientras que el Aprendizaje Automático evalúa el rendimiento de una técnica en términos de *capacidad de reproducir conocimiento* que ya se sabía, para la Minería de Datos la tarea principal es *descubrir conocimiento nuevo*. Respecto del nombre de Aprendizaje Automático, hay que tener en cuenta la diferencia de significado entre lo que se entiende por *aprender* para el caso de humanos y para el caso de máquinas. Así, en [43] se dan cinco definiciones de la palabra *aprender*, explicando que no todas ellas son aplicables a máquinas:

1. Adquirir conocimiento a través del estudio o la experiencia.
2. Ser consciente a través de la observación.
3. Memorizar.
4. Recibir información.
5. Ser instruido.

La posibilidad de aplicar las dos primeras definiciones de *aprender* a una máquina presenta ciertas dificultades. ¿Cómo saber si una máquina tiene conocimientos o si es consciente de algo? Sería posible evaluar la capacidad de una máquina de responder a

preguntas, pero esta capacidad no implicaría necesariamente un aprendizaje tal como se entiende para los humanos. En este contexto, la pregunta de si puede una máquina aprender se engloba dentro de otra pregunta más amplia: ¿puede una máquina razonar?. La pregunta anterior está relacionada con el nombre que recibe el área de conocimiento al que pertenece el Aprendizaje Automático, la Inteligencia Artificial. Dicho área estudia cómo crear máquinas que posean capacidad de razonar similar a la de los humanos. Para que una máquina sea considerada inteligente, debe pasar lo que se conoce como *Test de Turing*, consistente en realizar una serie de preguntas a una máquina y a un humano, y mostrar las respuestas a un juez humano. Se considera que la máquina es inteligente si el juez no es capaz de distinguir qué respuestas son del humano y cuáles de la máquina [160]. El test de Turing ha generado polémicas filosóficas en la comunidad científica [205]. Por un lado, ciertos autores defienden la ideonidad del test para probar la inteligencia de las máquinas, mientras que otros científicos afirman que una máquina podría pasar el test de Turing sin tener inteligencia, sino simplemente simulando el comportamiento de un humano. Así, un ordenador podría ejecutar un conjunto de instrucciones que hicieran que respondiera correctamente a una serie de preguntas, pero sin que los símbolos internamente significasen nada para el ordenador, por lo que no podría considerársele inteligente. Así, para el filósofo John Rogers Searle mientras que los programas informáticos tienen un carácter formal (sintáctico), la mente humana tiene contenidos (significado, semántica) [205].

Las tres últimas acepciones de aprender sí son más fácilmente verificables en máquinas. Sin embargo, resultan insuficientes porque no recogen que el aprendizaje debe resultar beneficioso para la realización de la tarea que se aprende [43]. Por ello, Chakrabarti et al. proponen la siguiente definición para el caso de máquinas:

- Cambiar el comportamiento de forma que mejore el rendimiento al realizar una acitividad.

Entones, al hablar de Aprendizaje Automático, aprender significa mejorar el rendimiento con la experiencia. Por ello, es necesario definir una función que evalúe el rendimiento del aprendizaje. Dentro de las técnicas de Aprendizaje Automático, en el presente estudio se analizan solamente aquellas que sirven para el Reconocimiento de Patrones. En este caso, el Aprendizaje Automático está íntimamente ligado a la optimización, ya que muchas técnicas se formulan como la *minimización de un riesgo* que compara los valo-

res predichos con los verdaderos valores que se desean predecir [116]. Según se incluya o no una penalización de la complejidad del modelo de predicción se habla de *minimización de riesgo empírico* (*empirical risk minimization*, en la bibliografía inglesa) y la *minimización del riesgo estructural* (*structural risk minimization*, en la bibliografía inglesa). La conveniencia de dicha penalización se detalla en el apartado 3.4.3. Sin embargo, de forma resumida se puede explicar que el error que se desea minimizar no debe referirse únicamente a los datos con los que se ha aprendido el modelo de predicción, sino a datos futuros. En otras palabras, se debe desarrollar un modelo con capacidad de generalización.

Existe otra diferencia entre el concepto de *aprendizaje* utilizado en Minería de Datos con respecto del habitual para personas. Dado que el modelo se aprende en base a una serie de ejemplos, se asume qué este puede cometer errores, ya que podrían presentarse nuevos ejemplos con informaciones no contempladas anteriormente. Entonces, no se busca que todas las predicciones sean correctas, sino que con alta probabilidad la mayoría de ellas sean correctas. Se habla de que el aprendizaje no es totalmente correcto sino *probablemente aproximadamente correcto* (*PAC learning*, en la bibliografía inglesa). Expresando la probabilidad de éxito como $1-\delta$ y el grado de corrección como $1-\varepsilon$, se dice que un algoritmo aprende un concepto en la forma PAC si puede hacer que los valores de δ y ε sean arbitrariamente pequeños [31]. Si existe una técnica de aprendizaje que pueda aprender un concepto en tiempo polinomial respecto al número de instancias, se dice que dicho concepto es PAC-aprendible. Las teorías del aprendizaje PAC determinan que el número mínimo de muestras para que una técnica de aprendizaje que reproduzca hipótesis en un espacio H pueda aprender unos conceptos $C \subseteq H$ es una función de la complejidad del espacio de hipótesis y de los inversos de δ y ε. Respecto de la complejidad del modelo, se distingue entre espacios de hipótesis finitos, donde la complejidad viene dada por el número total de hipótesis, e infinitos, en los que la complejidad se expresa a partir de la *dimensión VC* (dimensión de Vapnik-Chervonenkis, *VC dimesion* [138], en la bibliografía inglesa), el mayor número de puntos que una hipótesis de H puede separar (por ejemplo, 2 para una recta).

En la terminología usada a la hora de distinguir los tipos de técnicas usadas en el análisis de Minería de Datos, se distingue entre métodos descriptivos y predictivos [140]. Las técnicas descriptivas buscan condensar la información de grandes volúmenes de datos haciéndola más fácilmente interpretable, mientras que en las técnicas predictivas el hecho

de que los modelos construidos tengan mayor o menor complejidad no es lo más relevante, sino su capacidad de determinar valores de las variables estudiadas. Así, podemos categorizar las técnicas de Minería de Datos en el siguiente esquema:

- Análisis descriptivo
  - Segmentación de datos o *clustering*
  - Análisis de correlación
  - Reglas de asociación
  - Análisis de dependencia
- Análisis predictivo
  - Clasificación
  - Regresión
  - Análisis de Series Temporales

En la presente Tesis Doctoral, se han usado técnicas de clasificación y regresión, que son introducidas en 3.2.

Los principales campos de aplicación de la Minería de Datos son:

- Evaluaciones en el sector financiero [210]. Por ejemplo, a la hora de conceder un crédito o una hipoteca, o para elegir entre varias posibles opciones de inversión en la bolsa.
- Reconocimiento de imágenes [71], como el usado para determinar el número de matrícula de alguien que ha superado el límite de velocidad.
- Predicción de demanda industrial, por ejemplo en el sector eléctrico [214].
- Diagnóstico médico [14].
- Campañas de publicidad y ventas [17].
- Automatización industrial. Determinación de los parámetros de control óptimos en procesos de alta complejidad [81].

3.2. Tipos de problemas estudiados

Se han abordado dos tipos de problemas de predicción: la clasificación y la regresión. En ambos casos, el objetivo es encontrar un modelo que sea capaz de predicir una variable de salida, denominada *clase* en la terminología de Minería de Datos, en función de unas variables de entrada, denominadas *atributos*. Para ello se tiene un conjunto de datos con una serie de *instancias*, cada elemento de información compuesto por los atributos y la clase. Por ejemplo, al estudiar el comportamiento de los aerogeneradores, la clase representa si la máquina funciona correctamente o hay algún fallo, y los atributos las características que definen el nivel de vibración a través del remuestreo angular del espacio transformado de Fourier. En ese caso, cada instancia es cada dato registrado para el que se ha medido el par (atributos, clase).

Formalmente, si llamamos x a los atributos e y a la clase, tanto en clasificación como en regresión se obtiene una función f que predice los valores de y a partir de los de x, de la forma:

$$y = f(x) \tag{3.1}$$

Sin embargo, mientras que en los problemas de regresión la clase toma valores continuos (por ejemplo, la longitud de una pieza fabricada mediante tecnologías láser), en clasificación la clase sólo puede tomar una serie de valores discretos (por ejemplo, para los aerogeneradores, una clase representa el funcionamiento correcto y las otras siete los correspondientes siete tipos de fallos).

En la presente Tesis Doctoral se han analizado los siguientes problemas:

- En el estudio del micro-fresado 3D, un problema de regresión.
- En el estudio del pulido láser, un tipo de clasificación conocida como *ordinal*.
- En el estudio del mantenimiento de aerogeneradores,
  - un problema de clasificación general
  - un problema de clasificación conocida como *desequilibrada*.

Las peculiaridades de los tipos de problemas de predicción analizados en la *clasificación ordinal* y *clasificación desequilibrada* pueden consultarse en las Secciones 3.6 y 3.7, respectivamente.

3.3. Estadística Inferencial y Contraste de Hipótesis

3.3.1. Concepto de Contraste de Hipótesis

A través de la Estadística Inferencial se busca deducir propiedades válidas para la totalidad de una población sin necesidad de analizar todos sus individuos, sino solamente una parte de la población, a la que se denomina muestra [30]. La cualidad de la población que se desea estudiar se considera una variable aleatoria.

Para ilustrar estos tres conceptos básicos de la Estadística Inferencial (población, muestra y variable aleatoria), se toma como ejemplo una empresa que desea estudiar la fiabilidad de un determinado modelo de máquina. La población estaría formada por todas las máquinas de ese tipo, mientras que la muestra serían solamente aquellas sobre las que recogieran datos para analizar la fiabilidad. Como variable aleatoria podrían tomarse varias opciones, por ejemplo, el tiempo medio entre dos fallos en cada máquina.

De entre los campos de estudio de la Estadística Inferencial, en el presente apartado se explican los Contrastes de Hipótesis, ya que son claves para entender el aporte de la Tesis Doctoral frente a estudios anteriores. Otras áreas importantes son el Muestreo (selección de la muestra representativa del conjunto de la población) y la Estimación de Parámetros de la variable aleatoria.

El Contraste de Hipótesis es una técnica por la cual se verifica si una suposición sobre la población estudiada (llamada Hipótesis Nula, H_0) es compatible con la muestra a partir de la cual se analiza dicha población [30], o por el contrario dicha hipótesis debe ser rechazada (teniéndose en ese caso la Hipótesis Alternativa, H_1). Para ello, se busca que la Hipótesis Alternativa sea evidente desde un punto de vista estadístico, esto es, que tenga una alta probabilidad. El valor que tiene que alcanzar la probabilidad de la Hipótesis Alternativa para ser aceptada se conoce como nivel de confianza, p, siendo habitual expresar su valor en función de su probabilidad complementaria, α, el nivel de significación ($p = 1 - \alpha$). En el área de la Minería de Datos suele tomarse un nivel de confianza del 95 % ($\alpha = 0{,}05$), siendo también habituales valores del 90 % y 99 % ($\alpha = 0{,}1$ y $\alpha = 0{,}01$)

en otros ámbitos de ingeniería.

Es importante resaltar que en un Contraste de Hipótesis puede rechazarse la Hipótesis Nula, pero no probarse. En [30] se explica esta peculiaridad tomando como analogía un juicio, en el que se tiene que decidir si un acusado es inocente (Hipótesis Nula) o culpable (Hipótesis Alternativa). Se dice que "el acusado es inocente hasta que se demuestre lo contrario". En el juicio se puede probar la culpabilidad del acusado (rechazar la Hipótesis Nula, probar la Hipótesis Alternativa) o no hacerlo, lo que no supone probar la inocencia (probar la Hipótesis Nula). Por ello, las propiedades de la población que se considera interesante probar se formulan habitualmente en la Hipótesis Alternativa. Por ejemplo, se considera nuevamente el estudio de la fiabilidad de máquinas de una empresa. Si se desea probar que el tiempo entre fallos es mayor que una determinada cantidad, $t > t_0$, la Hipótesis Nula reflejará la situación opuesta, $t <= t_0$. En la Figura 3.1 se muestran cuáles serían las probabilidades de H_0 y H_1, suponiendo que el tiempo entre fallos siguiera una distribución normal.

En las ecuaciones 3.2 y 3.3 se indican las condiciones que cumplen las probabilidades de H_0 y H_1 en caso de aceptación y rechazo de la Hipótesis Nula.

$$\begin{cases} \text{Aceptación } H_0 \rightarrow \text{No hay evidencias de } H_1 \\ p(H_1) \leq 1-\alpha \\ p(H_0) \geq \alpha \end{cases} \tag{3.2}$$

$$\begin{cases} \text{Rechazo } H_0 \rightarrow \text{Evidencias de } H_1 \\ p(H_1) > 1-\alpha \\ p(H_0) < \alpha \end{cases} \tag{3.3}$$

En el estudio se han usado Contrastes de Hipótesis para comparar el grado de acierto de los predictores construidos con las técnicas de Minería de Datos. De este modo, es posible determinar si un algoritmo es más adecuado que otro para modelar un problema industrial, al considerar como Hipótesis Nula que los dos predictores tienen el mismo porcentaje de acierto. Si puede rechazarse dicha hipótesis hay diferencias significativas entre ambas técnicas de predicción.

Específicamente, el Contraste de Hipótesis usado es el test *t* remuestreado corregido [137]. Dicho test se formula a partir de la distribución *t* de Student, derivada de la distri-

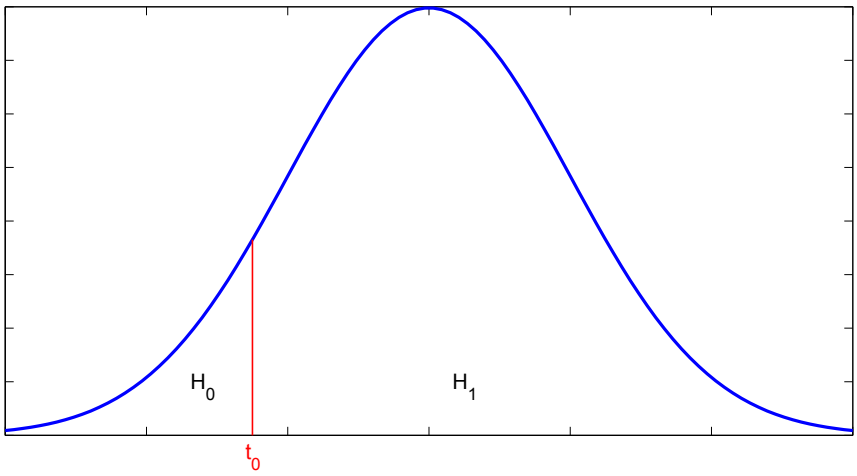


Figura 3.1: Áreas de probabilidad asociadas a las hipótesis de un contraste

bución Normal. En los siguientes apartados se introducen ambas distribuciones, pasando posteriormente a describir el test para comparar la precisión de dos predictores.

3.3.2. El Teorema del Límite Central y la Distribución Normal

Como se ha explicado en el apartado anterior, para realizar un Contraste de Hipótesis es necesario calcular la probabilidad de la Hipótesis Nula. Para realizar dicho cálculo es necesario considerar que la variable estudiada sigue una determinada distribución de probabilidad. El Contraste de Hipótesis usado en el presente estudio asume que las variables estudiadas, las precisiones de predictores según técnicas de Minería de Datos, pueden modelarse en base a la distribución Normal.

La validez de esta asunción viene dada por el Teorema del Límite Central, que establece que tanto la media como la suma de un conjunto suficientemente grande de variables aleatorias independientes e idénticamente distribuidas sigue una distribución normal, aunque cada una de ellas no siga individualmente dicha distribución. Así, estudiando para una población la variable aleatoria, X, la media y la suma de una muestra aleatoria suficientemente grande siguen una Distribución Normal. En la práctica se considera la Distribución Normal una buena aproximación a partir de 100 muestras [159].

3.3.3. La Distribución t de Student

Para comparar la precisión de dos predictores, expresada a través de las variables aleatorias X_1 y X_2, se usa una variable auxiliar que se construye como la resta de ambas estimaciones, $d = X_1 - X_2$. Entonces, el Contraste de Hipótesis que prueba si la precisión de los dos predictores es diferente es aquel que toma como Hipótesis Nula que la media de d sea nula ($\mu_0 = 0$). Se asume que la variable d sigue una Distribución Normal de varianza desconocida. Aunque es común en Estadística estimar la varianza de una población (S) a través de la varianza de una muestra (σ_d^2), en este caso se tiene la limitación del bajo número de muestras (el número total de repeticiones en una validación cruzada es pequeño). Por ello, no se considera adecuado aproximar el cociente entre d y σ_d por una normal estándar (de media 0 y varianza 1). En lugar de ello, se define en este contexto la *distribución t de Student*, que no sólo depende de d y σ_d, sino que además tiene un grado de libertad, k. En la ecuación 3.4 se muestra la expresión del estadístico usado para los test de comparación de dos predictores, y para n muestras sigue una distribución t de Student de $k = n - 1$ grados de libertad.

$$T = \frac{d - \mu_0}{\sqrt{\sigma_d / n}} = \frac{d}{\sqrt{\sigma_d / n}} \tag{3.4}$$

3.4. Validación de predictores

La validación es el proceso por el cual se estima el grado de precisión que alcanzan los modelos construidos. El objetivo buscado es cuantificar el acierto esperado para los datos no vistos en el entrenamiento del modelo, es decir, ver si las reglas proporcionadas por los modelos a partir de un conjunto de instancias son generalizables a cualquier otro grupo de instancias del problema analizado.

En este contexto, es importante resaltar el hecho de que siempre sería posible alcanzar una precisión muy alta si ésta se evalúa sobre los datos usados para construir el modelo, sin más que memorizar todas las instancias, pero con este procedimiento no se podría estimar la posible validez del modelo para predecir la clase de otros datos en el futuro. Por ello, en la validación se distingue entre el conjunto de datos con el que se construye el modelo, *datos de entrenamiento*, y los datos usados para medir la precisión de los clasificadores, los *datos de test*.

La forma más sencilla e intuitiva de hacer una validación es la conocida como método de retención (*holdout method*), en la que a partir de los datos iniciales se construye un único conjunto de entrenamiento y un único conjunto de test, tomando una partición aleatoria de los datos. Hay que tomar dos determinaciones: por un lado, qué porcentajes de instancias se usan para el entrenamiento y el test, y, por otro, qué método usar para obtener la división aleatoria. Los resultados experimentales muestran que esta técnica de validación es poco precisa, ya que los resultados dependen de la forma de hacer la partición de los datos [63]. Por ello surgió la validación más usada actualmente, la *validación cruzada*, en la que se utilizan varios subconjuntos para entrenamiento y test, de forma que todas las instancias en unos casos pertenecen a los conjuntos de entrenamiento y en otros a los de test [107].

Existen varios tipos de validación cruzada, pero la más habitual es la de "k repeticiones de N grupos" (*fold*, en la bibliografía inglesa), $k \times N$. En este procedimiento, los datos iniciales se dividen en k veces en N subconjuntos. Por cada repetición se evalúa el predictor N veces (una por cada subconjunto), resultando en un total de $k \times N$ evaluaciones. Para ello, cada uno de los subconjuntos sirve como datos de test en una única evaluación, formándose los datos de entrenamiento a partir de los $N-1$ subconjuntos restantes. Es recomendable que a la hora de formar las divisiones de los grupos se respete en éstas las proporciones entre las distintas clases, diciéndose entonces que la validación cruzada es *estratificada*. Las validaciones más usadas en Minería de Datos son las **validaciones cruzadas estratificadas** 10×10 [203] **y** 5×2 [58]. La validación 5×2 es más ligera computacionalmente si el tiempo de cómputo de entrenamiento es mayor que el tiempo de predicción, lo cual es habitual en la mayoría de técnicas de predicción. Por ello, se consigue ahorrar tiempo de experimentación empleando validaciones 5×2 cuando se tienen grandes volúmenes de datos. Sin embargo, la precisión de los clasificadores puede ser menor con validaciones 5×2 que con validaciones 10×10 si los conjuntos tienen un número limitado de instancias, por lo que en ocasiones puede ser más recomendable utilizar estas últimas. Este hecho es debido a que cada modelo se entrena con el 90 % de los datos iniciales en las validaciones 10×10, frente al 50 % de las validaciones 5×2.

En la Figura 3.2 se muestra un esquema de ejemplo de una validación cruzada estratificada 10×10, para un problema con dos clases, indicadas con círculos azules y verdes. La clase marcada en azul es tres veces más numerosa que la indicada en verde, proporción

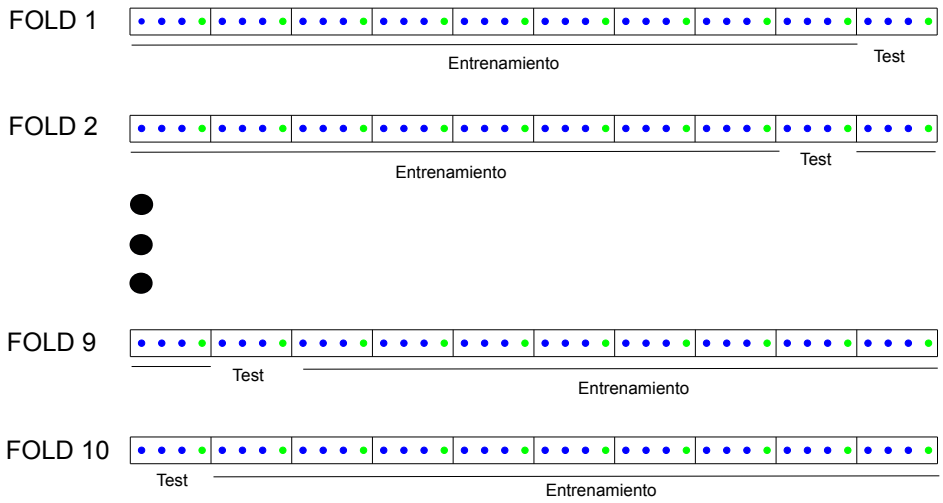


Figura 3.2: Esquema de una de las repeticiones de una validación cruzada $k \times 10$

que se respeta en cada uno de los grupos hechos en cada *fold*, representados por un rectángulo. En la figura sólo se indica una de las 10 repeticiones, para el resto de repeticiones se vuelve a hacer una división en grupos respetando la proporción entre clases.

3.4.1. El test *t* remuestreado corregido

Nadeau y Bengio [137] estudiaron cuál debía ser la formulación del estadístico usado para el Contraste de Hipótesis en el caso de comparar la precisión de dos clasificadores a través de una validación cruzada. Su análisis concluyó que en vez de usar directamente la varianza de la muestra (σ_d^2) para estimar la varianza de la población (S), era más adecuado usar un factor de corrección que tuviera en cuenta la correlación existente entre los conjuntos de entrenamiento de cada repetición de la validación cruzada. En la ecuación 3.5 se indica la reformulación de la estimación de la varianza, siendo k es el número de repeticiones de la validación cruzada, n_1 el número de instancias de entrenamiento y n_2 el de test. La expresión del estadístico usada en el test con la corrección de la varianza es la indicada en la ecuación 3.6.

$$s/n = \sigma_d^2 \left(\frac{1}{k} + \frac{n_2}{n_1} \right) \tag{3.5}$$

$$T = \frac{d}{\sqrt{\sigma_d^2 \left(\frac{1}{k} + \frac{n_2}{n_1}\right)}} \tag{3.6}$$

3.4.2. Medidas de precisión y funciones de pérdida

Si se desea cuantificar el rendimiento de un predictor es necesario establecer una medida que determine en qué grado se parece el valor de la clase que dicho predictor asigna con su valor real. Como se ha explicado al introducir el concepto de validación, una vez que se ha entrenado el predictor con el conjunto de entrenamiento, se dispone de un conjunto de test con una serie de instancias para las que se compara el valor real de la clase con el valor que pronostica el predictor a partir de los atributos de dichas instancias.

Entonces, el punto de partida para medir la precisión de una predicción es el mismo para los dos tipos de predicciones estudiadas, regresión y clasificación: comparar valores reales y predichos de la clase. Sin embargo, hablar de cuál es "el error de un predictor" es más complejo en un problema de clasificación que en un problema de regresión. La magnitud en la que medir el error resulta clara para predicción, al poder tomar directamente la variable predicha. Por ejemplo, imaginemos un regresor que se usa para predecir las dimensiones de un taladro resultantes de un proceso de fabricación con láser. Si la profundidad real del taladro fuera 10 μm y la predicha 8 μm, el error de profundidad sería de 2 μm. Por contra, para el caso de clasificación la variable predicha es nominal y no puede usarse directamente para cuantificar el error. Tal sería el caso de otro proceso de fabricación para el cual lo que se estudiara fuese el grado de rugosidad superficial de la pieza tratada, pudiendo tomar los valores Baja, Media, Alta. En este caso, podría tenerse un valor real de la rugosidad "Alta" y un valor predicho de "Media", no siendo tan inmediato cuantificar el error como en el ejemplo de predecir una longitud.

Ante la dificultad de no poder usar directamente la magnitud de la clase para cuantificar el error, se han propuesto dos alternativas diferentes para medir la precisión de un clasificador. La solución más sencilla consiste en tomar el porcentaje de instancias correctamente clasificadas, pero en este contexto surge también el concepto de *función de pérdida* (*loss function*, en la bibliografía inglesa). Dicha función toma como entradas dos variables escalares, los valores reales y predichos de la clase, y devuelve un valor escalar que es más pequeño cuánto más se parecen las dos entradas a la función [64]. Entonces,

llamando l a la función de pérdida, y_r al verdadero valor de la clase, y_e al valor estimado, el error cometido por un clasificador sobre un conjunto de validación con n instancias, e_V, viene dato por la ecuación 3.7.

$$e_V = \frac{1}{n}\sum_{n} l(y_r, y_e) \tag{3.7}$$

La función de pérdida más intuitiva y extendida para clasificación es la "pérdida 0/1", pero ésta no es convexa, por lo que no es conveniente su uso con algunas técnicas de aprendizaje basadas en minimización, que se explican en la Sección 3.5 [215]. Dicha función toma el valor 0 si la instancia analizada está correctamente clasificada y 1 en caso contrario.

El concepto de función de pérdida está relacionado con el de margen, que se explica en el Apartado 4.3. Dicho concepto se ha utilizado como justificación teórica de algunos clasificadores binarios (analizados en 3.5), pero su formulación para problemas de clasificación con múltiples clases es aún un tema de investigación abierto cuando se plantean los algoritmos en términos de margen [55], por lo que los trabajos que analizan funciones de pérdida para clasificación distintas de 0/1 suelen referirse únicamente a problemas binarios.

También es válida la idea de función de pérdida para el caso de regresión, siendo en este caso las opciones más habituales el error absoluto y error cuadrático medio, así como algunos derivados de los mismos, como la raíz del error cuadrático medio.

3.4.3. Elección de la complejidad de un predictor

Construir un modelo de predicción tratando de ajustarse al máximo a los datos de entrenamiento podría llevar a que dicho predictor tuviese poca capacidad de generalización. Para ilustrar este hecho se toma como ejemplo un problema de regresión polinómica en dos variables, la variable independiente, x, y la dependiente, y, con los datos que se muestran en la Figura 3.3, en la que cada dato se indica mediante un cuadrado azul.

En la Figura 3.4 vemos los ajustes de los datos que harían modelos polinómicos de distinto grado, indicados por las líneas en rojo. En la figura, para los grados 2 y 4 se tiene un ajuste razonable de los datos, mientras que el orden 1 es inadecuado por ser excesivamente sencillo y el orden 6 por ser excesivamente complejo. En la notación de

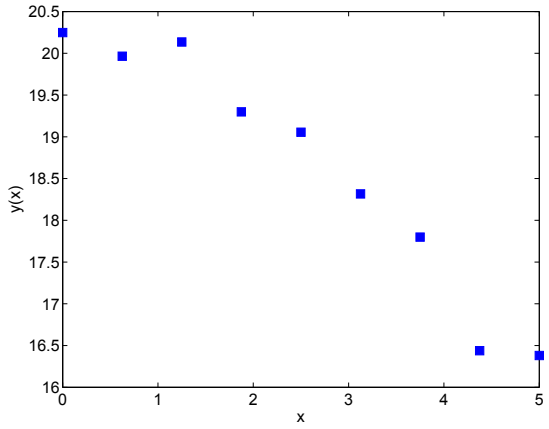


Figura 3.3: Muestras de ejemplo – regresión polinómica

Minería de Datos, se dice que para el orden 1 se tiene un problema de *falta de ajuste* (*underfitting*, en la bibliografía inglesa) y para el orden 6 un problema de *sobreajuste* (*overfitting*, en la bibliografía inglesa) [82]. Mientras que para el orden 1 podemos ver como el modelo (la línea roja) está alejado de algunas muestras, para el caso de tener orden 6 el modelo tiene una excesiva variabilidad, no representativa de los datos. Para ilustrar este segundo problema, se ha tomado la Figura 3.5, en la que se hace un nuevo ajuste de los datos, pero sin considerar las dos muestras indicadas por cuadrados verdes. Se pueden ver las predicciones que hacen los modelos de estas dos muestras para cada orden a través de los puntos rojos en la figura. Para el orden 6 el valor real y el predicho de las muestras están distanciados, debido al sobreajuste del modelo.

Es posible cuantificar numéricamente la pérdida de precisión que la falta de ajuste y el sobreajuste causan en los predictores, usando una Validación Cruzada. En el gráfico 3.6 se muestra el error total de predicción al variar el grado del polinomio, comparando la cantidad de error medida sobre el propio entrenamiento con la que estima la validación cruzada. En la figura se observa que es posible reducir el error sobre los datos de entrenamiento sin más que aumentar la complejidad del modelo, pero no sucede lo mismo en el caso de una validación cruzada [20].

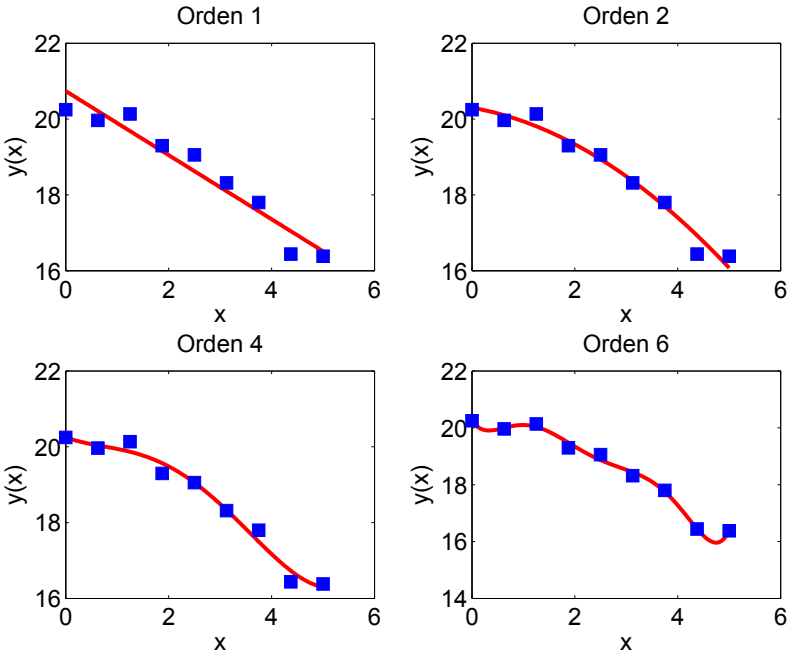


Figura 3.4: Ajustes polinomiales de las muestras de ejemplo

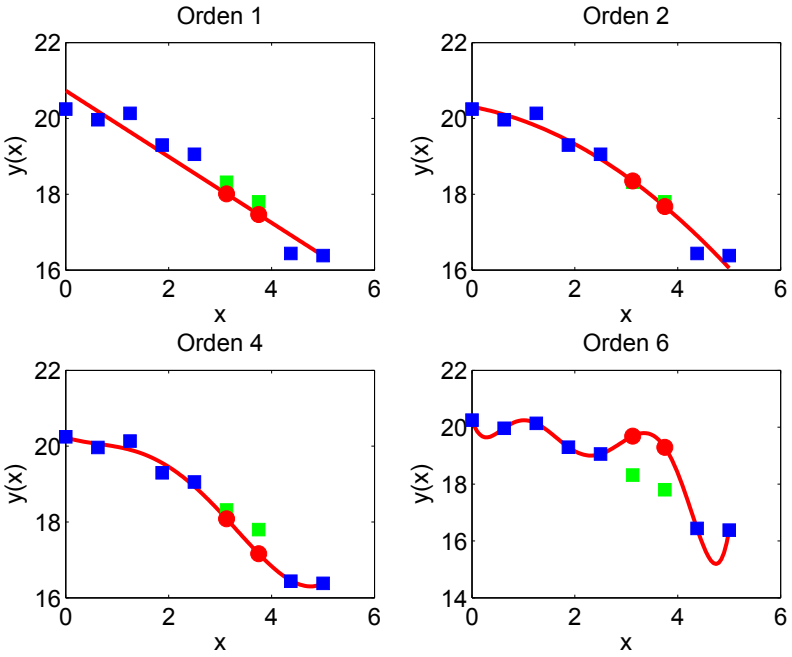


Figura 3.5: *Underfitting* y *overfitting*

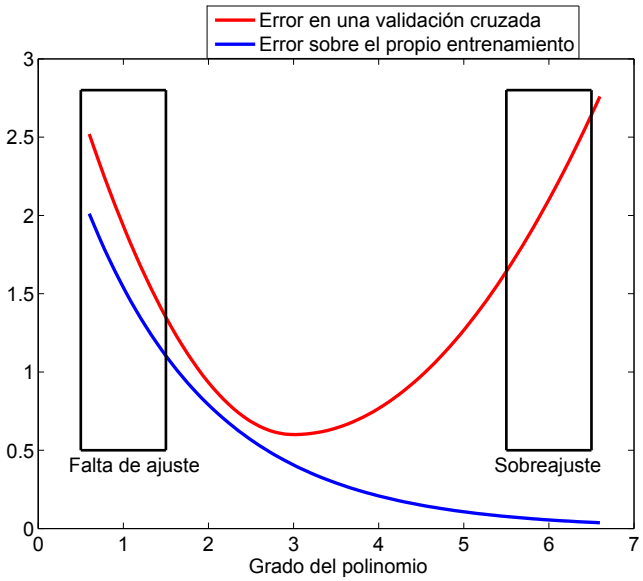


Figura 3.6: Evolución del error con falta de ajuste y sobreajuste

3.5. Familias de Predictores

Dentro de la Minería de Datos, pueden distinguirse cinco familias principales de técnicas de predicción [110]:

- Aprendizaje Inductivo [4]: Árboles [148] y Reglas [106] de decisión.
- Modelos probabilísticos o bayesianos [160]
- Ensembles [112]
- Modelos basados en funciones, destacando los SVMs (siglas del inglés *Support Vector Machine*) [26] y las Redes neuronales [13]
- Predicción en base a la similitud con respecto a las instancias almacenadas en el entrenamiento, conocida como "basada en instancias" (*Instance-based learning*, en la bibliografía inglesa [2])

A continuación se explica de forma general cada una de las estrategias de predicción, pudiéndose consultar cada técnica más detalladamente en el Capítulo de publicaciones (capítulo 5), así como en las referencias incluidas en las secciones correspondientes.

3.5.1. Aprendizaje inductivo

El aprendizaje inductivo busca extraer, a partir una serie reducida de ejemplos observados, reglas que describan de forma general la relación entre los atributos y la clase para toda la población [148]. Se distinguen dos categorías diferenciadas en los métodos que siguen este tipo de aprendizaje [4]: métodos de *divide y vencerás* (*Divide-and-Conquer*, en la bibliografía inglesa) o árboles de decisión y métodos de *separa y vencerás* (*Separate-and-Conquer*) o algoritmos de *covering*. La diferencia entre ambas categorías está en el número de clases tenido en cuenta cuando se construyen las reglas de decisión del modelo. Mientras que en los árboles se tienen en cuenta todas las clases, en los métodos de *covering* cada decisión se centra en un única clase, sin tener en cuenta las demás [4], buscando las combinaciones de atributos únicos para una clase. De entre la primera categoría los algoritmos más populares son ID3, C4.5 [148] y CART [29], mientras que la segunda categoría viene representada por los sistemas de extracción de reglas [106].

La introducción a los métodos de aprendizaje inductiva realizada en este apartado se centra en el árbol de clasificación tomado como referencia en el presente trabajo: C4.5 [148]. En este método se hacen divisiones sucesivas considerando un único atributo, buscando que las divisiones formadas tiendan a ser puras (i.e.; predomine una determinada clase entre las instancias que engloba) y difieran entre sí en el valor de esa clase. El proceso de división continúa de forma iterativa hasta que en las divisiones del último nivel jerárquico (llamadas *hojas*) se tienen todas las instancias de la misma clase o ya no quedan atributos en los que se pueda ramificar.

Existen varios índices para definir el grado de separación respecto a la clase que aporta cada división, siendo el más habitual la entropía. Este concepto deriva de *información asociada a un evento aleatorio*, idea introducida por Shannon en 1948 [173]. El problema que estudió entonces no fue el de la elección de reglas para clasificadores de tipo árboles de decisión, sino el problema de transmisión de información. Para ello, analizó cuál era el volumen de información que transmitía un mensaje, permitiendo estimar el número de bits necesarios para su comunicación. Aunque con diferente objetivo inicial, sus estudios son también útiles en este campo. De hecho, se ha indicado que se desean buscar decisiones que generen divisiones lo más homogéneas posibles. Pero este objetivo puede replantearse en términos de información, ya que cuanto más homogéneas sean las divisiones, menor será la cantidad de información necesaria para describirlos.

Shannon concluyó que la información que aporta un suceso es más relevante a medida que es menos probable, proporcionando por tanto menos información cuanto más ocurra de forma cotidiana. A partir de este hecho, definió la información asociada a un evento aleatorio. Para ello, buscó una medida que dependiera del inverso de la probabilidad, pero que además debía cumplir ciertas propiedades que la hicieran coherente con la teoría de probabilidad. Específicamente, hay que tener en cuenta que la probabilidad de que se produzcan conjuntamente dos sucesos independientes viene dada por el producto de sus probabilidades. Entonces, era necesario que la información total aportada por dos eventos independientes fuese coherente con la probabilidad resultante de la sucesión de eventos. En el esquema 3.8 se muestran las dos condiciones que cumple la medida de información asociada a un evento aleatorio (I), supuesto que se tienen dos eventos independientes, con probabilidades p_1 y p_2:

$$\begin{cases} I = f\left(\frac{1}{p}\right) \\ evento_1 \to I_1, evento_2 \to I_2 \Rightarrow evento_1 \text{ y } evento_2 \to I_1 + I_2 = f\left(\frac{1}{p_1 \times p_2}\right) \end{cases} \tag{3.8}$$

El logaritmo en cualquier base del inverso de la probabilidad es la formulación que cumple las dos propiedades anteriores (3.9).

$$I = log\left(\frac{1}{p}\right) = -log\,(p) \tag{3.9}$$

Una vez que se dispone del índice I asociado a un evento aleatorio, se aplica este concepto a una decisión usada en un clasificador. Para ello, se define la entropía, S, que es la media ponderada del índice I de cada uno de los eventos asociados a la decisión. Si se tienen N eventos, S viene dada por la expresión 3.10.

$$S = \sum_{i=1}^{N} p_i I_i = \sum_{i=1}^{N} -p_i log\,(p_i) \tag{3.10}$$

Algunos tipos de árboles de clasificación, como ID3, seleccionan directamente como mejor regla de decisión aquella que minimiza la entropía, es decir, aquella hace una división tal que los grupos que se forman tienen un volumen de información más pequeño (en los grupos se tenga la máxima homogeneidad posible, lo más próximo posible a tener sólo

una clase por grupo). Sin embargo, ID3 sólo maneja atributos nominales, por lo que se definen otros tipos de árboles como C4.5, que emplean conceptos asociados a la entropía, y no la entropía directamente, que pueden ser extendidos a atributos numéricos, como se explica posteriormente. Estos conceptos surgen al comparar cuánta información proporciona un árbol de decisión antes y después de añadir una regla. Entonces, se definen dos magnitudes a partir del de *entropía*, la *ganancia de información*, G y el *ratio de ganancia*, R. En las ecuaciones 3.11 y 3.12, se muestra el cálculo de dichos valores cuando la comparación es la información inicial del conjunto de entrenamiento, E, frente a la información tras incorporar una regla de división según el atributo X.

$$G(E,X) = S(E) - S(E,X) \tag{3.11}$$

$$R(E,X) = \frac{G(E,X)}{S(X)} \tag{3.12}$$

A continuación se muestra un ejemplo de como elegir una regla de decisión en un predictor tipo árbol. Para ello, se toman como punto de partida los datos de una máquina que tiene tres sensores de nivel (S1, S2, S3) para detectar los posibles tipos de fallos que se pueden producir (eléctrico, E, y mecánico, M). Cada sensor tiene tres posibles niveles, el nivel de alarma (A), el nivel intermedio (I) y el caso en el que la magnitud medida no llega al nivel suficiente, con el sensor desactivado (D). Las siete instancias de ejemplo para construir el árbol se muestran en el Cuadro 3.1. Usando la expresión 3.10 se puede calcular la entropía de cada una de las posibles ramas que se pueden formar al tomar divisiones según los tres atributos (cada valor concreto de cada uno de los tres sensores en este caso). Los sumatorios tienen dos términos en este ejemplo, según la probabilidad de las clases M y E. Se han escogido logaritmos en base dos para el cálculo, con lo que la entropía queda medida en bits, pero como base del logaritmo podría tomarse cualquier base, como se ha indicado anteriormente. En el Cuadro 3.2 se muestran la distribución de la clase según cada posible división en los atributos, las probabilidades de las clases asociadas a dichas divisiones, p_M y p_E y la entropía resultante.

A partir de los valores de entropía que se acaban de mencionar (entropía en cada rama, valores del Cuadro 3.2), se calcula la entropía asociada a una regla de división promediando todas sus ramas, tal como se indica en 3.13.

Cuadro 3.1: Instancias de ejemplo para elegir una regla de un árbol de decisión

S1	S2	S3	Tipo
D	D	I	M
D	D	A	M
A	D	I	M
A	A	D	M
D	A	D	E
D	A	A	E
A	A	A	E

$$\begin{cases} S(E,S1) = \frac{4}{7} \times 1 + \frac{3}{7} \times 0{,}918 = 0{,}965 \\ S(E,S2) = \frac{3}{7} \times 0 + \frac{4}{7} \times 0{,}811 = 0{,}464 \\ S(E,S3) = \frac{2}{7} \times 1 + \frac{2}{7} \times 0 + \frac{4}{7} \times 0{,}918 = 0{,}810 \end{cases} \tag{3.13}$$

La división según los valores del segundo sensor proporciona una entropía menor, por lo que dicho sensor sería escogido como primera regla de división del árbol en el caso de utilizar el método ID3. En el conjunto de ejemplo hay tres instancias de clase E y cuatro de clase M, por lo que las probabilidades del modelo sin ninguna regla de decisión son $p_E = 3/7$ y $p_M = 4/7$, lo que resulta en una entropía de los datos de entrenamiento de $S(E) = 0{,}985$ bits. La entropía asociada a los dos primeros sensores sin tener en cuenta la clase, sino sus propios valores, $S(X)$, es coincidente, $S(S1) = S(S2) = 0{,}985$ bits, ya que los dos tienen dos únicos posibles valores (D y A), con probabilidades 3/7 o 4/7. Para el sensor S3, sin embargo, que también puede tomar el valor I, la entropía es $S(S3) =$ 1,557 bits. La ganancia de información que aporta cada sensor viene dada por la resta de $S(E)$ y cada uno de los tres $S(E,X)$ indicados en 3.13. Dividiendo dichas ganancias por los respectivos valores de $S(X)$, se tienen los ratios de ganancia: $G(E,S1) = 0{,}020$, $G(E,S2) = 0{,}529$, y $G(E,S2) = 0{,}112$. Para este ejemplo C4.5 tomaría la misma regla que ID3, ya que para el sensor S2 además de tener menor entropía se tiene mayor ratio de ganancia.

En el ejemplo anterior, los atributos pueden tomar únicamente valores discretos. C4.5 tiene un mecanismo para transformar los atributos numéricos continuos a dos valores discretos a partir de un umbral, dividiendo los valores entre aquellos que son menores o iguales al umbral y aquellos que son mayores. Para ello, primero ordena los valores del

Cuadro 3.2: Elección de la mejor regla de un árbol de decisión

S1		S2		S3		
S1=D	S1=A	S2=D	S2=A	S3=D	S3=I	S3=A
M	M	M	M	M	M	M
M	M	M	E	E	M	E
E	E	M	E			E
E			E			
$p_M = \frac{1}{2}$	$p_M = \frac{2}{3}$	$p_M = 1$	$p_M = \frac{1}{4}$	$p_M = \frac{1}{2}$	$p_M = 1$	$p_M = \frac{1}{3}$
$p_E = \frac{1}{2}$	$p_E = \frac{1}{3}$	$p_E = 0$	$p_E = \frac{3}{4}$	$p_E = \frac{1}{2}$	$p_E = 0$	$p_E = \frac{2}{3}$
$S = 1$	$S = 0{,}918$	$S = 0$	$S = 0{,}811$	$S = 1$	$S = 0$	$S = 0{,}918$

Cuadro 3.3: Posibles umbrales para un atributo numérico en C4.5

Atributo	20	20	30	30	30	40	40
Clase	M	M	E	E	E	M	M

atributo y después considera como posibles umbrales el promedio de dos valores para los que se produce un cambio de clase [149]. Para cada posible umbral, define un índice de calidad a partir del ratio de ganancia visto en 3.12, y un término que penaliza la complejidad del modelo en términos de cuál es el número de valores distintos del atributo que se desea discretizar, N. Siendo $|E|$ el número de valores totales del atributo que se desea discretizar, el índice de calidad a optimizar, IC, tiene la expresión indicada en 3.14. Entonces, C4.5 compara $R(E,X)$ para cada atributo nominal con $IC(E,X)$ para cada atributo numérico continuo y posible umbral seleccionando la división con mayor valor de dichas magnitudes.

$$IC(E,X) = R(E,X) - \frac{log_2(N-1)}{|E|} \tag{3.14}$$

Continuando con el ejemplo de 3.1,podría considerarse un cuarto atributo numérico, que en $|E| = 7$ instancias de entrenamiento sólo toma $n = 3$ valores distintos, 20, 30 y 40, como se muestra en el Cuadro 3.3. Sólo hay dos cambios de clase para el atributo numérico, entre 20 y 30, lo que da un umbral de 25, o entre 30 y 40, lo que da un umbral de 35. Entonces, para dichos umbrales C4.5 calcularía el ratio de ganancia como se ha mostrado para el ejemplo con atributos nominales, pero penalizaría dicho valor con $\frac{log_2(2)}{7} = 0{,}1429$.

En los ejemplos que se han mostrado anteriormente se tenía un problema de clasificación, esto es, la clase podía tomar solamente una serie de valores discretos. También existen árboles de decisión para problemas de regresión, en los que la clase toma valores numéricos continuos. Entonces, no es válido medir la variabilidad de la clase según la división en unos atributos a partir de la entropía sino que hay que usar otros índices definidos para variables continuas. Una posibilidad es usar la desviación típica, *sd*. Por ejemplo, el método M5 [150] maximiza la reducción de error estimada, $\Delta(e)$, definida a partir de la desviación típica, el número total de instancias del conjunto a dividir, E, y el número de instancias donde el atributo analizado para formar la división toma su valor i-ésimo de entre los N posibles, E_i, de acuerdo con la ecuación 3.15.

$$\Delta(e) = sd(E) - \sum_{i=1}^{N} \frac{|E_i|}{|E|} sd(E_i) \tag{3.15}$$

Hay dos tipos de árboles de decisión para regresión: los *árboles de regresión* y los *árboles de modelos*. La diferencia entre ambos tipos reside en cómo se calcula el valor predicho para cada hoja del árbol a partir de las instancias de la misma [150]. Mientras que un *árbol de regresión* se toma el promedio de la clase en las instancias de la hoja, para los *árboles de modelos* se construye un modelo de regresión lineal en cada hoja.

Poda de árboles de decisión

Los árboles construidos por este método pueden generar modelos excesivamente complejos que sobreajusten los datos [130]. Por ello, es habitual reducir la complejidad de los árboles de decisión en lo que se conoce como *poda*. Se usan dos sistemas de poda, la "pre-poda" o "poda" a secas (*pre-pruning* o *pruning*, en la bibliografía inglesa) y la "post-poda" (*post-pruning*, en la bibliografía inglesa). La pre-poda se aplica como criterio de parada durante el crecimiento del árbol, mientras que la post-poda se aplica a posteriori, después de que el árbol haya sido construido. Existen varios sistemas de pre y post poda. A continuación se explican las técnicas de poda empleadas por el método de clasificación C4.5, probablemente el método inductivo más popular en la bibliografía de Minería de Datos. Para este método concreto, la pre-poda se basa en un test chi-cuadrado y la post-poda se conoce como *colapso del árbol* (*collapse*, en la bibliografía inglesa).

En la pre-poda se hace una estimación del error cometido con cada nodo que se va

añadiendo al árbol, desestimando un posible nuevo nodo y parando el crecimiento del árbol cuando el error no es menor de forma estadísticamente significativa al aumentar el tamaño del árbol.

En [148], Quinlan propuso considerar únicamente válidas aquellas ramificaciones del árbol que originasen subconjuntos cuya distribución de clases fuese diferente de forma estadísticamente significativa a la que se obtendría de forma teórica con una distribución aleatoria. Para ello, se hace uso de lo que se conoce en Estadística como *test de independencia chi-cuadrado*. Dicho test sirve para determinar si hay diferencia entre dos variables cualitativas (aplicado a problemas de clasificación, entre las clases del problema) en una muestra dividida en varias poblaciones (cada población se corresponde con un subconjunto formado por la ramificación del árbol para el caso de la poda).

El estadístico usado en el test, χ^2, se construye a partir de los cuadrados de las diferencias entre las frecuencias reales y las esperadas de cada clase, como muestra la ecuación 3.16. Se considera que el estadístico sigue una distribución de probabilidad conocida como *chi-cuadrado*, lo que da nombre al test. Dicha distribución tiene un parámetro, k, el número de grados de libertad del problema estudiado. Para el caso de analizar, para un problema de c clases, la posible poda de una ramificación en h hojas, resulta un valor de $k = (h-1)(c-1)$. Entonces, tal como se explicó al introducir los contrastes de hipótesis (Apartado 3.3.1), es posible comparar el valor del estadístico obtenido con un valor umbral a partir de un nivel de significación, α.

Para ilustrar el funcionamiento del test de independencia chi-cuadrado cuando se usa para pre-poda, imaginemos un problema de clasificación con dos clases, A y B, y una ramificación en dos hojas, la izquierda, I, y la derecha, D, con un total de m instancias. Los variables p y n indican los números reales de instancias de cada clase en cada subconjunto, mientras que p' y n' denotan los números teóricos (esto es, los valores esperados para los números de instancias de cada clase con una distribución aleatoria). Entonces, se construirían dos tablas con los valores reales y teóricos, tal como se muestra en el esquema 3.4, resultando la expresión indicada en 3.16. En este caso, una ramificación en dos y en un problema con dos clases, el número de grados de libertad de la distribución chi-cuadrado sería $k = (2-1)(2-1) = 1$. Tomando un nivel de significación del 5 % ($\alpha = 0{,}05$), resulta un valor umbral de $\chi^2_{\alpha=0{,}05,k=1} = 3{,}8415$. Por tanto, C4.5 incluiría la rama solo si el valor de χ^2 calculado según 3.17 fuese mayor que 3,8415.

Cuadro 3.4: Esquema de un test chi-cuadrado para pre-poda

Valores reales

	A	B	Total
I	p_I	n_I	$T_I = p_I + n_I$
D	p_D	n_D	$T_D = p_D + n_D$

Valores teóricos

	A	B
I	$p'_I = p \cdot T_I / m$	$n'_I = n \cdot T_I / m$
D	$p'_D = p \cdot T_D / m$	$n'_D = n \cdot T_D / m$

$$\chi^2 = \sum_{subconjuntos} \frac{(\text{real} - \text{teórico})^2}{\text{teórico}} \tag{3.16}$$

$$\chi^2 = \frac{(p_I - p'_I)^2}{p'_I} + \frac{(n_I - n'_I)^2}{n'_I} + \frac{(p_D - p'_D)^2}{p'_D} + \frac{(n_D - n'_D)^2}{n'_D} \tag{3.17}$$

En el colapso se busca reducir la complejidad de un árbol ya construido manteniendo una precisión similar en el diagnóstico. Para ello se analiza el error de predicción esperado, comparando cuál sería el nivel sin podar el árbol y podándolo, procediendo a eliminar aquellas ramas que no bajen el error esperado. La probabilidad de error en una rama del árbol es la media ponderada de la probabilidad de error en cada una de sus hojas, cuya estimación es necesaria. Dicha estimación es habitualmente *pesimista*, es decir, se toma la máxima probabilidad de error de acuerdo con las instancias de cada hoja de árbol.

Para ello, se considera que el número de errores del clasificador al predecir la clase viene dado por una distribución de Bernoulli. Esta distribución de probabilidad discreta sirve para calcular la probabilidad de tener un determinado número de éxitos, x, al hacer una secuencia de n ensayos de Bernoulli (i.e., en los que sólo se pueden obtener dos resultados, *éxito* o *fracaso*) con probabilidad de éxito p en cada uno de ellos. Se dice entonces que x sigue una distribución binomial de parámetros n y p, $x \sim B(n,p)$.

El problema de encontrar la máxima probabilidad de error para un conjunto de datos (en este caso, las instancias de cada hoja del árbol) según la distribución binomial, se corresponde con tomar el límite superior del Intervalo de Confianza del parámetro p de la distribución binomial $B(n,p)$. Al igual que se explicó en el Apartado 3.3.1 para los Contrastes de Hipótesis, los Intervalos de Confianza dependen del nivel de significancia, $(1-\alpha)$. De esta forma, cuanto más certeza se quiera tener de que el valor del parámetro a calcular está en un intervalo más grande ha de ser este. Para el cálculo del parámetro p sue-

len usarse en la bibliografía de Minería de Datos aproximaciones al intervalo de confianza de dicho parámetro basadas en el teorema del límite central que toman la distribución normal. La solución de dicho intervalo de confianza se basa en la solución de una ecuación cuadrática, resultando la expresión indicada en 3.18 [134], donde z_α es la inversa de la función de distribución de una normal estándar evaluada en $1-\alpha$. La expresión anterior es función del valor esperado del parámetro p, $\hat{p}$, que para el caso de la poda de los árboles es el tanto por uno de errores en la hoja de árbol analizada, $\hat{p} = x/n$.

$$p \in \left(\frac{\hat{p} + \frac{z^2_{\alpha/2}}{2n} - \frac{z_{\alpha/2}}{\sqrt{n}} \sqrt{\hat{p}(1-\hat{p}) + \frac{z^2_{\alpha/2}}{4n}}}{1 + \frac{z^2_{\alpha/2}}{n}}, \frac{\hat{p} + \frac{z^2_{\alpha/2}}{2n} + \frac{z_{\alpha/2}}{\sqrt{n}} \sqrt{\hat{p}(1-\hat{p}) + \frac{z^2_{\alpha/2}}{4n}}}{1 + \frac{z^2_{\alpha/2}}{n}} \right) \tag{3.18}$$

El valor usual del nivel de significancia empleado en Minería de Datos para la poda de árboles es $\alpha = 0{,}5$, es decir, $z_{\alpha/2}$ es el valor que hace que la probabilidad acumulada de una distribución normal sea el 75 % ($z_{\alpha/2} = 0{,}6745$). A continuación se ilustra el proceso de post-poda con el ejemplo de la Figura 3.7, en el que se representa el análisis del funcionamiento de una máquina como "bueno" (B) o "malo" (M), en función de 3 variables, $x1$, $x2$ y $x3$. En la rama con 3 hojas marcada con el recuadro en azul, se clasificarían como buenas a las instancias en la hoja $x3 = Bajo$ y como malo a las instancias en las otras dos hojas. Entonces, el valor esperado en cada una de las 3 hojas, $\hat{p}$, y el valor superior del intervalo de confianza para la estimación del verdadero valor del parámetro, p, serían los indicados en 3.19. Haciendo la media ponderada, resulta un valor de 0,6663, tal como se indica en 3.20. Este valor se compara con el que resultaría de juntar las 3 hojas de la rama en una única hoja. En ese caso tendríamos tantas instancias buenas como malas ($M = B = 5$), resultando $\hat{p} = 0{,}5$ y $p = 0{,}6043$. La probabilidad de error al colapsar la rama es mayor que la probabilidad sin colapsar la rama, por lo que no se podaría a dicha rama.

$$\begin{cases} x3 = Bajo \rightarrow \hat{p} = \frac{1}{4}, \;\; p = 0{,}4162 \\ x3 = Medio \rightarrow \hat{p} = \frac{1}{2}, \;\; p = 0{,}7152 \\ x3 = Alto \rightarrow \hat{p} = \frac{1}{4}, \;\; p = 0{,}4162 \end{cases} \tag{3.19}$$

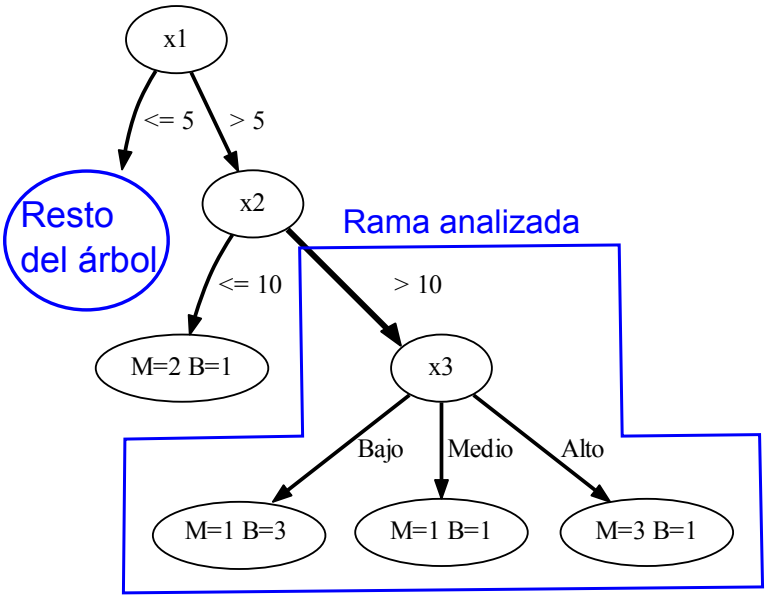


Figura 3.7: Ejemplo de post-poda

$$p = \frac{4}{10} \cdot 0{,}4162 + \frac{2}{10} \cdot 0{,}7152 + \frac{4}{10} \cdot 0{,}4162 = 0{,}6663 \tag{3.20}$$

3.5.2. Predicción bayesiana

Generalmente, este tipo de sistemas de predicción se han considerado más adecuados para problemas de clasificación y no de regresión [67], por lo que en el presente estudio se han usado únicamente como clasificadores. En este caso, las técnicas bayesianas consideran la pertenencia a una clase como una variable aleatoria. De esta forma, clasifican a una instancia como de la clase que maximice la probabilidad de la clase condicionada a que los atributos tengan los valores de la instancia a clasificar [97]. El cálculo de dicha probabilidad condicionada viene dada por el teorema de Bayes, lo que da nombre a la familia de clasificadores. Denotando por C a la clase sobre la cual se quiere estimar la probabilidad, x a la instancia de test, e indicando el operador "|" condicionalidad ("$a|b$", *a condicionado b*), el teorema expresa el cálculo de probabilidades según la expresión 3.21.

$$p(C|x) = \frac{p(C) \cdot p(x|C)}{p(x)} \tag{3.21}$$

A la hora de comparar probabilidades de las distintas clases, el denominador, $p(x)$, es constante, por lo que no influye en la maximización. Por otro lado, el cálculo del término $p(C)$, probabilidad de una clase, es inmediato, ya que se estima a partir de las frecuencias de entrenamiento. Entonces, de los tres términos que aparecen en la expresión del teorema de Bayes, es únicamente $p(x|C)$ el que exige tiempo de cálculo. La forma de estimar dicho término difiere entre los distintos tipos de técnicas bayesianas. Se explica a continuación el procedimiento para el caso del clasificador Naïve Bayes (bayesiano ingenuo), que toma como hipótesis la independencia de los atributos. Esta suposición permite que $p(x|C)$ pueda ser calculado mediante el producto de la probabilidad de cada atributo, x_i, condicionada a la clase (3.22).

$$p(x|C) = p(x_1|C) \cdot \ldots \cdot p(x_n|C) \tag{3.22}$$

El cálculo de cada uno de los factores en el producto anterior depende del tipo de atributo considerado. Si es nominal es el porcentaje de instancias que tuvieron ese valor, mientras que si es numérico se estima suponiendo que la probabilidad sigue una distribución normal o gausiana, caracterizada por la media y la desviación típica de los datos en el entrenamiento.

3.5.3. Predicción basada en funciones

Los sistemas de predicción basados en funciones buscan establecer de forma explícita la relación analítica entre la clase a predecir y los atributos. Se distinguen para ello dos tipos de enfoques [183]. Por un lado, los métodos paramétricos suponen que el tipo de relación entre los atributos y la clase es conocida y puede expresarse en función de un número reducido de parámetros. Así, para plantear un métodos paramétrico es necesario establecer unas hipótesis previas sobre cómo se ve afectada la clase por cada atributo. Por ejemplo, si la variación respecto de un atributo concreto es lineal, cuadrática o de otro tipo. Por otro lado, los métodos no paramétricos no presuponen el tipo de relación entre los atributos y la clase. Dentro de los métodos no paramétricos resultan especialmente interesantes aquellas técnicas de predicción que permiten representar relaciones de cualquier

grado complejidad entre los atributos y la salida. Dichos predictores pueden ser explicados como sistemas de aproximación de funciones. En total, se analizan en este apartado cuatro técnicas de predicción basadas en funciones:

- Métodos paramétricos
  - Regresión lineal, como ejemplo de aplicación de métodos paramétricos a problemas de regresión.
  - Regresión logística, como ejemplo de aplicación de métodos paramétricos a problemas de clasificación.
- Aproximación de funciones de complejidad arbitraria
  - Redes neuronales.
  - Máquinas de vectores soporte.

A continuación se expone la formulación de cada una de las técnicas de predicción basadas en funciones consideradas. No se detallan todos los desarrollos matemáticos, que pueden consultarse en el libro de Bishop et al. [20].

Regresión lineal

En la regresión lineal se busca obtener una formulación sencilla de la clase a predecir, utilizando para ello simplemente una combinación lineal de los atributos, lo que facilita la comparación de la influencia de cada atributo individual. El valor estimado de la clase, y_e, se expresa como la suma de los productos de cada atributo por unos parámetros β_i, más un término independiente, β_0 [20], como se indica en la ecuación 3.23, para un problema con m atributos.

$$y_e = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_m x_m \tag{3.23}$$

La suma de productos de la ecuación 3.23 puede expresarse a través de un producto escalar de vectores, incorporando un atributo adicional auxiliar, x_0, de valor unidad. Entonces, definiendo los vectores columna $B = [\beta_0, \beta_1, ..., \beta_m]^T$ y $X = [x_0, x_1, ..., x_m]^T$ (denotamos con minúsculas a valores escalares, y con mayúsculas a valores vectoriales o

matriciales), de $m+1$ componentes, la predicción viene dada de forma simplificada por la ecuación 3.24.

$$y_e = B^T X \tag{3.24}$$

El valor óptimo del vector de parámetros B se obtiene minimizando el error cuadrático, E, entre la clase real, y, y la predicha, y_e. Suponiendo que se tienen n instancias y denotando al valor real de la i-ésima instancia por y_i y al de los atributos por X_i, los valores óptimos de B vienen dados por la ecuación 3.25.

$$E = \sum_{i=1}^{n} \left(y_i - B^T X_i\right)^2, \quad B_{optimo} = \underset{B}{\text{argmín}}\, E \tag{3.25}$$

El valor de B que minimiza E se obtiene igualando la derivada de E respecto de B a cero, como se indica en la ecuación 3.26. Es posible simplificar la expresión anterior usando notación matricial, al definir un vector columna Y cuyas componentes sean los valores reales de la clase para las n instancias y una matriz ϕ cuya transpuesta tenga por columnas los n vectores X_i ($\phi^T = [X_1|...|X_n]$, de dimensión $(m+1) \times n$). Entonces, la condición de óptimo se reduce a una igualdad de producto de matrices, obteniéndose la solución en 3.27, multiplicando a ambos lados por la inversa de los factores. Dada una matriz cualquiera, A, al producto $(A^T A)^{-1} A^T$ se le denomina como *matriz pseudoinversa de Moore-Penrose*, y se denota por A^+. Entonces, también se expresan los parámetros B óptimos respecto al error cuadrático como $\phi^+ Y$.

$$\frac{dE}{dB} = 0 \rightarrow \sum_{i=1}^{n} \left[y_i - B^T X_i\right] X_i^T = 0 \rightarrow \sum_{i=1}^{n} y_i X_i^T = B^T \sum_{i=1}^{n} X_i X_i^T \tag{3.26}$$

$$\phi^T Y = \left(\phi^T \phi\right) B \rightarrow B = (\phi^T \phi)^{-1} \phi^T Y = \phi^+ Y \tag{3.27}$$

La causa de que la matriz ϕ^+ reciba el nombre de *pseudoinversa* puede verse si se omite la presencia de error en la formulación. Así, imaginemos que fuera posible que el valor de la clase real coincidiera siempre con el de la clase estimada. Entonces, el valor estimado de la clase, visto en la ecuación 3.24, sería también el valor real de la clase, y, y esto sucedería para las n instancias (X_i, y_i) del problema. Por tanto, se satisfarían n ecuaciones escalares, $y_i = B^T X_i$. Es posible expresar estas n ecuaciones escalares en una

única ecuación matricial a partir de la matriz ϕ, como se indica en 3.28. Si la igualdad de la ecuación anterior fuese posible, B se obtendría mediante la inversa de ϕ, $B = \phi^{-1}Y$. Sin embargo, sólo las matrices cuadradas con determinante no nulo tienen inversa, por lo que en general ϕ no es invertible. En adelante nos referiremos a la igualdad planteada en 3.28 como *en ausencia de error*. Dicha ecuación facilita la comprensión tanto del significado de la *pseudoinversa* como de la formulación del *problema de optimización dual*, que se explica a continuación, y que es clave para los predictores de tipo *Máquinas de Vectores Soporte* (Apartado 3.5.3).

$$Y = \phi B \tag{3.28}$$

Como se ha explicado, la regresión lineal puede plantearse en ausencia de error como encontrar el vector B que multiplicado por la matriz ϕ resulte en el vector Y, $\phi B = Y$, viniendo entonces la solución dada por la pseudoinversa de ϕ, ϕ^+ ($B = \phi^+ Y$). Sin embargo, en ocasiones se prefiere representar el problema de optimización en base a otros parámetros, α, permitiendo observar la linealidad que existe entre B y ϕ. Se habla entonces de la *representación dual* del problema de optimización, en la que los nuevos parámetros a optimizar α están relacionados con los anteriores, B, según la ecuación 3.29. Es posible manipular la expresión de la pseudoinversa vista en 3.27 para demostrar que si $\left(\phi^T\phi\right)$ es invertible siempre va a existir un α que cumpla la relación de linealidad [174]. Así, en 3.30 se multiplica a la pseudoinversa por un término igual a la matriz identidad, $\left(\phi^T\phi\right)\left(\phi^T\phi\right)^{-1}$.

$$B = \phi^T \alpha \tag{3.29}$$

$$B = \left(\phi^T\phi\right)^{-1}\phi^T Y = \phi^T\phi\left(\phi^T\phi\right)^{-2}\phi^T Y = \phi^T\alpha,\ \alpha = \phi\left(\phi^T\phi\right)^{-2}\phi^T Y \tag{3.30}$$

Reemplazando en 3.28 B por su valor en función de α, mostrado en 3.29, se llega a que el problema de optimización dual depende de la matriz $G = \phi\phi^T$, que se conoce como *matriz de Gram* (3.31). Por tanto, el cambio a la representación dual permite que la obtención de los parámetros óptimos dependa exclusivamente de la matriz de Gram, eliminando dependencias de términos aislados en ϕ. Esta transformación no es ventajosa para

el caso ahora analizado, la regresión lineal en los atributos sin transformar, pero sí resulta útil cuando se aplican ciertas transformaciones sobre los atributos, X. Específicamente, los predictores de tipo *Máquinas de Vectores Soporte* con núcleos no lineales emplean la matriz de Gram, concepto sobre el que se volverá en el correspondiente apartado (Apartado 3.5.3).

$$Y = \phi\phi^T\alpha = G\alpha,\ G = \phi\phi^T \tag{3.31}$$

Regresión logística

Un tipo frecuente de problema de clasificación es la predicción de variables categóricas binarias, también llamadas *dicotómicas*, en los que la clase a predecir puede tomar dos valores opuestos, tales como la ausencia o presencia de una determinada característica en un individuo (donde la clase valdría SÍ o NO) [54]. La regresión logística surge al buscar una técnica de predicción para este tipo de problemas con una formulación sencilla como la vista para la regresión lineal. Dado que la clase a predecir no es numérica, no puede ser directamente la salida de una regresión lineal, por lo que en su lugar se utiliza la probabilidad de la clase SÍ, p.

Debido a que la probabilidad varía en el rango [0,1], mientras que la regresión lineal puede variar entre $-\infty$ y $+\infty$, no se usa la probabilidad directamente como salida de la regresión lineal, sino una transformación de la misma en el rango [$-\infty$,$+\infty$]. Esta transformación se construye a partir de lo que se conoce como *odds*, término inglés que en este contexto significa *posibilidades*, cuyo rango es [0,$+\infty$] y cuya formulación se muestra en 3.32. La regresión logística se define como una regresión lineal cuya salida es el logaritmo de *odds* (ecuación 3.33, donde se mantiene la notación vectorial para B y X usada al explicar la regresión lineal).

Para obtener el valor de p en función de B se toma la exponencial de 3.33, resultando la dependencia indicada en la ecuación 3.34. A la función que relaciona la salida de la regresión lineal, B^TX, con la probabilidad, p, se la denomina función *logística* y se la denota por *Logit*. De esta forma, se puede expresar p abreviadamente en función de *Logit* (ecuación 3.35). Esta función permite relacionar la regresión logística con la clasificación binaria mediante redes neuronales, como se verá más adelante.

$$odds = \frac{p}{1-p} \tag{3.32}$$

$$log(odds) = B^T X \tag{3.33}$$

$$\frac{p}{1-p} = e^{B^T X} \rightarrow p\left(1+e^{B^T X}\right) = e^{B^T X} \rightarrow p = \frac{e^{B^T X}}{1+e^{B^T X}} \tag{3.34}$$

$$Logit(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}},\ p = Logit\left(B^T X\right) \tag{3.35}$$

A diferencia de en la regresión lineal, en este caso no se considera al error cuadrático una medida del error adecuada. Por ello, el cálculo de los parámetros óptimos se hace en este caso mediante lo que se conoce como *maximizar la verosimilitud*. Este proceso consiste en considerar que cada una de las instancias es un evento aleatorio independiente, y encontrar aquellos valores de los parámetros para los cuales es más probable que justamente se haya producido la combinación de eventos aleatorios representados por las muestras. El tipo de evento aleatorio que representa el valor de la clase de cada instancia en el problema considerado por la regresión logística se conoce como *ensayo de Bernoulli*. En este tipo de experimentos aleatorios hay dos posibles resultados, *éxito* o *fracaso* (análogos a los valores de la clase SÍ y NO, respectivamente). Entonces, se dice que la probabilidad de éxito (o de la clase SÍ) es p y la probabilidad de fracaso (o de la clase NO) es $q = 1 - p$. Representado por $y = 1$ el éxito (la clase SÍ) y por $y = 0$ el fracaso (la clase NO), la función de probabilidad que devuelve p para $y = 1$ y q para $y = 0$ es 3.36.

La probabilidad de tener exactamente la combinación de valores de la clase para un conjunto de n instancias viene dada por el productorio de las probabilidades de la clase de cada una de las instancias, ya que la probabilidad de la unión de varios sucesos aleatorios es el producto de sus probabilidades. Entonces, la función de verosimilitud, V, es la indicada en 3.37.

El valor óptimo del parámetro p es aquél que maximiza V. Como en el caso de la regresión lineal, el óptimo se produce cuando la derivada vale cero. Sin embargo, la presencia del productorio hace que al derivar V directamente resulte una expresión complicada. Por ello se aplica el logaritmo, lo que transforma los productos en sumas, más sencillas de

derivar. Además se invierte el signo de la transformación resultante, pasando así de un problema de maximización a un problema de minimización. Entonces, se tiene una función de error, $E = -log(V)$, como se indica en la ecuación 3.38. Para el caso de que los logaritmos estén en base 2 a esta función de error se la conoce como *entropía cruzada* (*cross entropy*, en la bibliografía inglesa).

$$\text{prob[una instancia, } y] = p^y (1-p)^{1-y} \tag{3.36}$$

$$V = \text{prob[muestras, } y_i,\ i = 1..n] = \prod_{i=1}^{n} p^{y_i} (1-p)^{1-y_i} \tag{3.37}$$

$$-E = \sum_{i=1}^{n} y_i log(p) + (1-y_i)\, log\,(1-p) = \sum_{i=1}^{n} y_i log\left(\frac{p}{1-p}\right) + \sum_{i=1}^{n} log(1-p) \tag{3.38}$$

Es necesario expresar E en función de B para poder obtener su valor óptimo. El término que multiplica a y_i es justamente sobre el que se hace la regresión lineal, $log\left(\frac{p}{1-p}\right) = B^T X$, mientras que el término del otro sumatorio se expresa en función de los parámetros B según se indica en 3.39. Por tanto, la expresión de E en función de B es 3.40.

$$1 - p = 1 - \frac{e^{B^T X}}{1 + e^{B^T X}} = \frac{1 + e^{B^T X} - e^{B^T X}}{1 + e^{B^T X}} = \left(1 + e^{B^T X}\right)^{-1} \tag{3.39}$$

$$-E = \sum_{i=1}^{n} y_i B^T X - \sum_{i=1}^{n} log\left(1 + e^{B^T X}\right) \tag{3.40}$$

El problema de encontrar un mínimo de la función de error E no tiene solución exacta, sino que se resuelve de forma aproximada con métodos numéricos, tales como los *métodos de descenso*, que se explican en mayor detalle en el Apartado 3.5.5, y que requieren del cálculo de gradiente de E, $\nabla_B E$. Las componentes de dicho gradiente se obtienen derivando respecto a un componente cualquiera de los componentes de B, β_j, según se muestra en la ecuación 3.41. El gradiente se puede expresar de forma matricial en función de la matriz ϕ, vista para la regresión lineal, definiendo P e Y como vectores columna en los que las componentes i-ésimas son los valores de la probabilidad y la clase para la i-ésima instancia, p_i e y_i, como muestra la ecuación 3.42.

$$\frac{dE}{d\beta_j} = 0 \rightarrow \sum_{i=1}^{n} y_i x_j - \sum_{i=1}^{n} \frac{e^{B^T x_i}}{1 + e^{B^T x_i}} x_j = \sum_{i=1}^{n} (y_i - p) x_j = 0 \tag{3.41}$$

$$\nabla_B E = \phi^T (Y - P(B)) = 0 \tag{3.42}$$

En el trabajo de Bishop et al. [20], se explica una forma eficiente de hallar los parámetros en este caso mediante otro método numérico, el método de Newton-Raphson, para el que se necesita calcular, además de las derivadas del primer orden incluidas en el gradiente, derivadas de segundo orden, las componentes de la matriz conocida como *Hessiana*, H. H es el gradiente del gradiente, y en este caso puede expresarse a través de una matriz diagonal, R, y de X, como se muestra en 3.43.

$$H = \nabla_B \nabla_B E = X^T R X, \;\; R_{ii} = p_i(B)\,(1 - p_i(B)), \;\; R_{ij} = 0 \text{ si } i \neq j \tag{3.43}$$

Para problemas de clasificación binaria sólo es necesario establecer un modelo para calcular la probabilidad de una de las dos clases, ya que entre las dos probabilidades deben sumar 1. Análogamente, en un problema con K clases, determinando la probabilidad de $K-1$ clases la probabilidad restante viene dada restando a la unidad la suma de las demás. Sin embargo, en este caso se prefiere (por facilitar la formulación y el cálculo) plantear una regresión lineal para la probabilidad de cada una de las clases [126]. Esto hace que algunos de los parámetros sean redundantes, lo que se conoce como modelo *sobre-parametrizado* (*overparametrized model*, en la bibliografía inglesa).

A continuación se explica la regresión logística para un problema binario usando una regresión lineal por cada una de las dos probabilidades de las clases del problema, formulación que es generalizable de forma inmediata a cualquier número de clases. En este caso, para poder usar regresores lineales se realizan dos cambios respecto a cuando sólo había una regresión lineal:

- Se introduce un término, Z, para asegurarse de que las probabilidades sumen 1.
- Una vez que se ha introducido Z, no es necesario utilizar *odds*, basta con tomar el logaritmo de la probabilidad.

Los parámetros a hallar vienen dados por 3.44. Tomando la exponencial puede verse que cada ecuación es equivalente a $p_i = e^{B_i^T x}/Z$. Por tanto, se puede obtener Z sumando la

exponencial de ambas ecuaciones, como se indica en 3.45.

$$\begin{cases} ln(p_1) = B_1^T X - ln(Z) \\ ln(p_2) = B_2^T X - ln(Z) \end{cases} \tag{3.44}$$

$$p_1 + p_2 = 1 = \frac{e^{B_1^T X} + e^{B_2^T X}}{Z} \rightarrow Z = e^{B_1^T X} + e^{B_2^T X} = \sum_{i=1}^{2} e^{B_i^T X} \tag{3.45}$$

La expresión de cada una de las probabilidades, p_i, en función de los parámetros, B_i, se obtiene sustituyendo el valor de Z en 3.44. En 3.45 se ha deducido la expresión de Z con solamente $K = 2$ clases, pero dicha expresión es válida para cualquier K. Por tanto, en la regresión logística para K clases, la probabilidad de la clase j-ésima viene dada por la ecuación 3.46. Dicha ecuación se puede expresar de forma abreviada definiendo la función vectorial *Softmax* (ecuación 3.47, para un vector de K componentes), cuyo nombre no suele traducirse aunque su significado equivaldría a *valor suavizado de la función máximo*. Entonces, el vector de probabilidades, P, es la función $SoftMax$ de un vector Z cuyas componentes son la regresión lineal de los atributos con los parámetros correspondientes, $z_i = B_i^T X$ (ecuación 3.48). La función *Softmax* permite relacionar la regresión logística multiclase con la clasificación multiclase mediante redes neuronales, como se verá más adelante. *Logit* es un caso particular de *Softmax* para dos clases, fijando todas las componentes del vector B asociado a la clase fracaso a cero ($B = B_1,\ B_2 = 0$). La salida de *Softmax* es una normalización de las distintas exponenciales $[e^{B_1^T X}, ..., e^{B_m^T X}]$ a un rango válido para un vector de probabilidades, de forma que la suma de todas las componentes sea 1 y se mantengan las proporciones que había entre las diferentes componentes antes de normalizar. A cada una de las exponenciales se las denomina *puntuación* (*score*, en la bibliografía inglesa). Entonces, el nombre de *valor suavizado de la función máximo* viene dado porque a cada clase se le asigna una probabilidad proporcional a su puntuación, no como ocurriría tomando directamente el máximo (asignando probabilidad 1 a la clase con mayor puntuación y 0 al resto).

$$p_j = \frac{e^{B_j^T X}}{\sum_{i=1}^{K} e^{B_i^T B}} \tag{3.46}$$

$$Softmax(Z)_j = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}},\ j = 1..K \tag{3.47}$$

$$P = Softmax(Z),\ z_i = B_i^T x \tag{3.48}$$

Como en el caso binario, para encontrar los parámetros B_i óptimos se define una función de coste basada en la entropía cruzada. Mientras que para dos clases sólo se usaba la probabilidad, p, de la clase éxito ($y = 1$), por ser la probabilidad de la otra clase $1 - p$, para el caso multiclase se definen tantas probabilidades como clases, con lo que la entropía cruzada, S, se generaliza según la expresión 3.49. El operador $\{a = b\}$ se denomina función indicador, y vale 1 si $a = b$ y 0 en otro caso. Para el caso binario, bastaba con tomar y como indicador de la clase $y = 1$ y $(1 - y)$ como indicador de la clase $y = 0$. Ahora, sin embargo, no puede obtenerse una representación sencilla de las clases que permita obtener una función $p = p(y)$ como en el ensayo de Bernoulli, sin recurrir a la función indicador.

$$S = \sum_{i=1}^{n} \sum_{clase=1}^{K} \{y_i = clase\} log\left(p_{clase}\right) \tag{3.49}$$

Como en el caso binario, para encontrar los parámetros óptimos se busca minimizar la función de error $E = -S$. Sin embargo, en el caso multiclase el sobre-paremetrizado puede ocasionar problemas de convergencia a algunos métodos numéricos de optimización. Por ello, en ocasiones se define otra función de coste añadiendo un término que penaliza valores grandes de los parámetros, como se muestra en la ecuación 3.50, donde λ es una constante de compensación entre el término de penalización y la entropía, que debe fijarse antes de la minimización ($\lambda > 0$). La componente j-ésima del gradiente (respecto de la clase $y = j$) de la función de coste viene dado entonces según la expresión 3.51.

$$E = -\sum_{i=1}^{n} \sum_{clase=1}^{K} \{y_i = clase\} log\left(p_{clase}\right) + \frac{\lambda}{2} \sum_{clase=1}^{K} \sum_{j=0}^{m} \beta_{clase,j}^2 \tag{3.50}$$

$$\left[\nabla_B E\right]_j = -\sum_{i=1}^{n} \left[x_i \left(\{y_i = j\} - p_{clase\ j}\right)\right] + \lambda B_j \tag{3.51}$$

Aproximación de funciones de complejidad arbitraria

Tanto la regresión como la clasificación pueden verse como un problema de aproximación de funciones. Existen una serie de técnicas de Minería de Datos que utilizan este hecho para construir un modelo de predicción. Para ello, suponen que existe una función

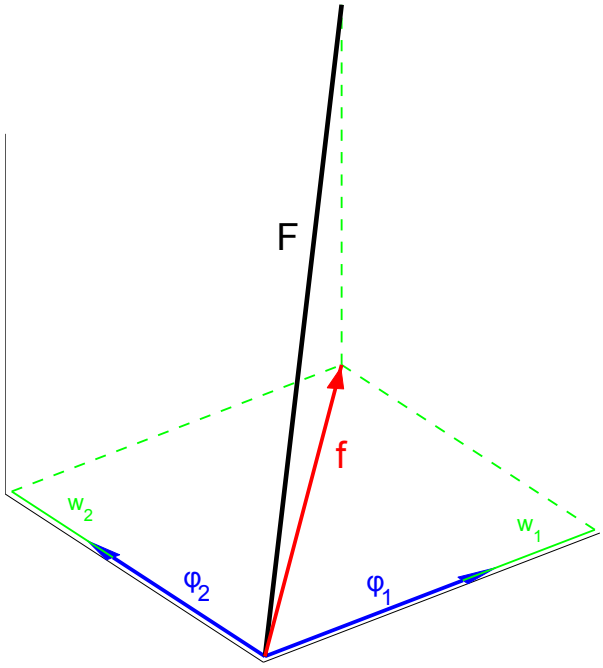


Figura 3.8: Aproximación de funciones

que relaciona los atributos de entrada con la salida. Dicha función puede tener cualquier grado de complejidad y es de formulación desconocida, por lo que se busca otra función que la aproxime a partir de los datos de entrenamiento.

El problema viene expresado matemáticamente por la ecuación 3.52. Siendo $f(x)$ la función que se desea aproximar, se construye un aproximador $\widehat{f}(x,w)$ a partir de la combinación lineal de N funciones elementales $\phi_i(x)$, cada una de ellas ponderada por un peso, w_i. Las funciones elementales, $\phi_i(x)$, generan un subespacio de funciones, pudiéndose ver la aproximación de la función, $\widehat{f}(x,w)$, como la proyección de $f(x)$ sobre dicho subespacio [3]. Una interpretación gráfica sencilla de este resultado es posible tomando únicamente dos funciones elementales ortogonales entre sí generando un plano como subespacio asociado, y una función en tres dimensiones, como se muestra en la Figura 3.8. En dicha figura, la función a aproximar se indica en negro y la aproximación en rojo, viniendo representadas las funciones elementales, ϕ_1 y ϕ_2, en azul, y los pesos, w_1 y w_2 en verde.

$$\widehat{f}(x,w) = \sum_{i=1}^{N} w_i \phi_i(x) \tag{3.52}$$

Las dos técnicas de predicción basadas en funciones con gran popularidad en la literatura de Minería de Datos son las Redes Neuronales Artificiales y las Máquinas de Vectores Soporte, SVMs. Ambos algoritmos constituyen aproximadores universales de funciones [79], esto es, no tienen restricciones respecto del tipo de funciones a las que pueden aproximar, característica que los diferencia de otros sistemas de aproximación, tales como las series de Fourier (que aproximan a funciones periódicas) o las series de Taylor (que sólo aproximan localmente a la función en un punto).

Aunque las redes neuronales y los SVMs tienen prestaciones similares, los planteamientos que dieron lugar a su desarrollo y su formulación son muy diferentes. Por un lado, el aprendizaje de los SVM tiene su origen en los trabajos sobre la teoría del aprendizaje estadístico [191]. Esta teoría considera que la forma de calcular los parámetros del modelo con otras técnicas de predicción está excesivamente centrada en los datos de entrenamiento, lo que puede llevar a una mala generalización, ya que es habitual minimizar los errores de predicción para el propio entrenamiento. Frente a ello, los SVMs [26] buscan predicciones con la máxima capacidad posible de generalización.

Al profundizar en los predictores de tipo SVM y las redes neuronales, se detallarán las diferencias entre ambos predictores, remarcando en este punto las similitudes que hacen que puedan englobarse dentro de la misma categoría, a través de la gran semejanza entre el tipo de decisiones generadas por los modelos de predicción. De hecho, puede comprobarse la equivalencia de ciertas arquitecturas de redes neuronales con determinados tipos de SVM. Tomando el caso de clasificación, las funciones que determinan las fronteras entre clases obtenidas con un SVM con kernel gaussiano son equivalentes a las proporcionadas por las redes neuronales de tipo "de base radial", y lo mismo sucede con los SVM de kernel sigmoide y una red neuronal de dos capas, sin capa oculta [95]. Esta equivalencia se refiere a la formulación del espacio de funciones generado, y no a que tengan que coincidir exactamente los parámetros de las funciones obtenidos para unos datos concretos de entrenamiento, ya que la forma de obtener estos parámetros en una red neuronal y un SVM es diferente. Puede consultarse una explicación más detallada de la formulación de las redes neuronales y los SVMs en los artículo "*Modelling laser milling of micro-cavities for the manufacturing of DES with ensembles*" (sección 5.1) y *An SVM-Based Solution for Fault Detection in Wind Turbines* (sección 5.2) incluidos en el capítulo 5.

Máquinas de vectores soporte (SVM)

En primer lugar se explica el funcionamiento de los predictores SVM para el caso de clasificación binario, analizando otros problemas de predicción más adelante. El objetivo del método es obtener fronteras de separación entre las clases lo más distantes en lo posible de las instancias más cercanas, lo que se conoce como *maximizar el margen*. En la Figura 3.10 se explica gráficamente el concepto maximizar el margen para un problema linealmente separable y considerando un tipo de SVM conocido como *lineal*. Se podrían definir infinitas rectas que separasen las dos clases, marcadas por los cuadrados y los círculos en la figura. Sin embargo, la recta marcada en continuo es mejor que las marcadas en discontinuo, porque deja más margen de separación para todas las instancias (es la recta a más distancia de sus instancias más próximas de cada clase).

Matemáticamente, se construye la frontera entre las dos clases a través de una función auxiliar, $g(x)$, cuyo valor es función de los atributos, x, a través de un vector de pesos asociados a cada atributo, w, y un término independiente de los mismos, b, como indica la ecuación 3.53, donde $< .. >$ denota el producto escalar. Representando a una clase por $y = 1$ y a la otra por $y = -1$, una instancia que hiciera $g(x) = 0$ estaría en la frontera, mientras que en otro caso la clasificación del SVM sería el signo de $g(x)$. Entonces, para el cálculo de la frontera que maximiza el margen, se definen otras dos rectas auxiliares paralelas a la misma, una por clase y pasando por las instancias más cercanas a la frontera ($g(x) = 1$ para $y = 1$ y $g(x) = -1$ para $y = -1$), como se muestra en la Figura 3.9. Las rectas paralelas están a una distancia $\frac{2}{||w||}$, por lo que maximizar la distancia es equivalente a minimizar $||w||$, sujeto a que $g(x)y \geq 1$. En la práctica pueden tenerse problemas linealmente no separables, en los que puede haber alguna instancia de una clase rodeada de instancias de la otra. Por ello, se introducen las *holguras*, ξ, que permiten que dichas instancias puedan ser incorrectamente clasificadas. El problema de optimización queda como se indica en la ecuación 3.54, donde C es un término de compensación entre la maximización del margen y la cantidad de error de clasificación tolerado mediante las holguras, y n el número de instancias.

$$g(x) =< w,x > +b \tag{3.53}$$

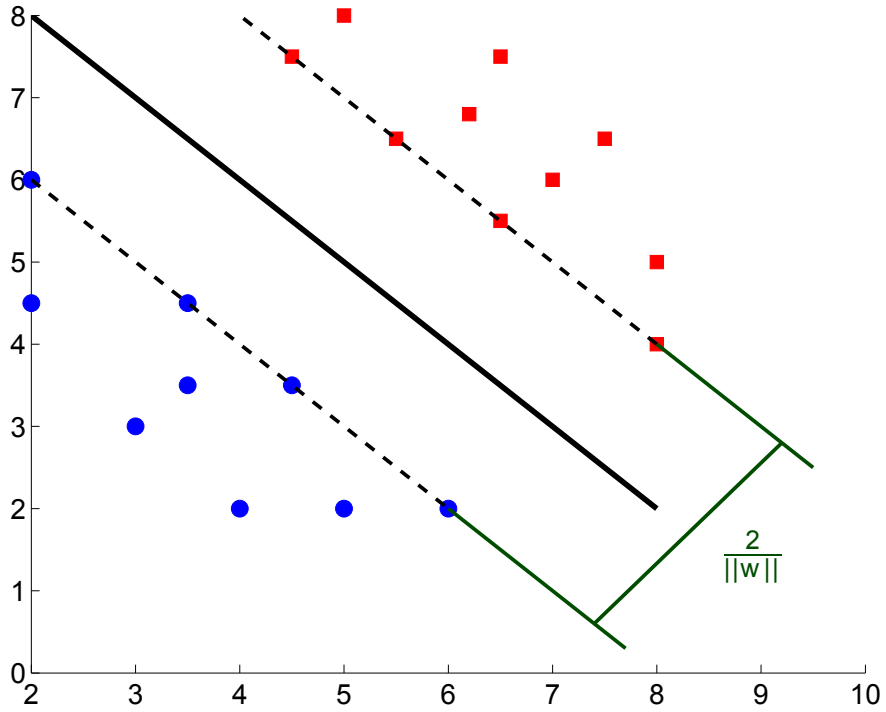


Figura 3.9: Optimización de la distancia de dos rectas paralelas en SVM

$$\begin{aligned} &argmin_{w,\xi,b}\{\tfrac{1}{2}||w||^2 + C\textstyle\sum_{i=1}^{n}\xi_i\} \\ &\text{sujeto a } \; g(x_i)y_i \geq 1-\xi_i, \;\; \xi_i \geq 0 \;\; i = 1...n \end{aligned} \tag{3.54}$$

Al igual que se mostró para la regresión lineal, el problema de optimización de un predictor SVM puede plantearse en términos de una formulación dual. Aunque el tipo de función a optimizar en este caso no es lineal, también se obtiene que los nuevos parámetros dependen de la matriz de Gram, G. En este caso, para llegar a la formulación dual se utiliza la técnica de las condiciones de Karush-Kuhn-Tucker, una generalización del método de los multiplicadores de Lagrange a problemas en los que existen restricciones de desigualdad. La técnica de los multiplicadores de Lagrange permite transformar un problema de optimización con restricciones únicamente de igualdad en uno nuevo equivalente sin restricciones. Para ello, se redefine una nueva función objetivo, llamada *Lagrangiano*, incorporando un nuevo término por cada restricción del problema en la formulación inicial. Dichos términos vienen dados por los productos de los *multiplicadores de Lagrange*, λ_k (que dan nombre al método) y las funciones implicadas en las restricciones, $r_k(x) = 0$.

Así, si la función objetivo del problema de optimización inicial es $h(x)$, el Lagrangiano, L, para un problema con N_r restricciones, viene dado por la expresión 3.55. En la optimización presente en los SVMs hay desigualdades, por lo que el *Lagrangiano* ha de tener restricciones. Definiendo el problema como una minimización de $h(x)$ y todas las restricciones como $r_k(x) \leq 0$, los multiplicadores de Lagrange están sujetos a ser valores positivos.

$$L(x,\lambda) = h(x) + \sum_{k=1}^{N_r} \lambda_k \left(r_k(x)\right) \tag{3.55}$$

Aplicando la técnica de las condiciones de Karush-Kuhn-Tucker a 3.54, se obtiene el Lagrangiano para el predictor SVM. Para ello, en primer lugar es necesario expresar las restricciones en la forma estándar, $r_i(x) \leq 0$, como se indica en 3.56. El Lagrangiano que se obtiene tiene la expresión de la ecuación 3.57. Entonces, los valores de w y b óptimos se obtienen igualando a cero la derivada del Lagrangiano respecto de dichos parámetros (3.58).

El valor del Lagrangiano dual se obtiene al reemplazar en L el valor óptimo de w. Puede comprobarse que se elimina la dependencia de b, w, ε y α. La expresión final del Lagrangiano es 3.60, solamente dependiente de λ, y donde G_{ij} son las componentes de la matriz de Gram, explicada para la regresión lineal [20].

$$\begin{cases} y_i g(x_i) \geq 1-\varepsilon_i \rightarrow 1-\varepsilon_i - y_i\left(< w, x_i > +b\right) \leq 0 \\ \varepsilon_i \geq 0 \rightarrow -\varepsilon_i \leq 0 \end{cases} \tag{3.56}$$

$$L(w,b,\varepsilon,\lambda,\alpha) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\varepsilon_i + \sum_{i=1}^{N}\lambda_i\left[1-\varepsilon_i - y_i\left(< w, x_i > +b\right)\right] + \sum_{i=1}^{N}\alpha_i\left(-\varepsilon_i\right) \tag{3.57}$$

$$\begin{cases} \frac{dL}{dw} = w - \sum_{i=1}^{N}\lambda_i y_i x_i = 0 \rightarrow w = \sum_{i=1}^{N}\lambda_i y_i x_i \\ \frac{dL}{db} = -\sum_{i=1}^{N}\lambda_i y_i = 0 \\ \frac{dL}{d\varepsilon_i} = C - \lambda_i - \alpha_i = 0 \rightarrow \lambda_i = C - \alpha_i \end{cases} \tag{3.58}$$

$$||w||^2 = < w, w > = \sum_{i=1}^{m}\lambda_i y_i < w, x_i > = \sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_i\lambda_j y_i y_j x_i^T x_j \tag{3.59}$$

$$L(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \lambda_i \lambda_j y_i y_j, \;\; G_{ij}\, G_{ij} = x_i^T x_j \tag{3.60}$$

Los parámetros óptimos, λ, del Lagrangiano dual proporcionan una solución dispersa: para la mayoría de los puntos $\lambda_i = 0$, contribuyendo únicamente al sumatorio aquellos puntos que están en la frontera de decisión o dentro de la franja dada por las variables de holgura, ε. A estos únicos puntos cuya λ_i es no nula se les llama *vectores soporte*.

La ecuación 3.60 se ha obtenido utilizando los atributos originales del problema de clasificación considerado, sin aplicarles ninguna transformación. Sin embargo, dicha ecuación permite obtener la formulación de los SVMs en un espacio transformado de los atributos, teniendo en cuenta la estructura de la matriz de Gram. Los componentes de dicha matriz no dependen explícitamente de los valores de los atributos, x, sino de sus productos escalares. La importancia de este hecho reside en que el producto escalar de dos vectores tiene un significado geométrico en términos de similaridad o distancia, por lo que la matriz de Gram es una matriz de similaridades o distancias entre las instancias de entrenamiento [169]. Mientras que para el SVM lineal se parte de las distancias en el espacio original de los atributos, para otros tipos de SVM también se toma la matriz de Gram, pero midiendo las distancias en un espacio transformado. Esto se consigue mediante lo que se conoce como el *truco del kernel* (*kernel trick*, en la bibliografía inglesa). Este sistema permite obtener la matriz de Gram en el espacio transformado sin tener por ello que transformar los atributos, utilizando una función *kernel* que proporciona directamente la distancia entre dos puntos en un espacio transformado. La interpretación geométrica del producto escalar para el espacio original puede verse a partir de la ecuación 3.61, que muestra el valor de dicho producto para dos vectores, x_1 y x_2, donde α es el ángulo entre dichos vectores. Es el coseno la parte de la expresión relacionada con la similaridad geométrica, ya que mide cómo se parecen las direcciones de ambos vectores. Además, se dice que el producto escalar *induce una norma*, ya que se puede definir la norma de un vector a partir del producto escalar de dicho vector por sí mismo (ecuación 3.62). Por tanto, la distancia de dos vectores en un espacio transformado puede calcularse a partir de la norma de su diferencia en dicho espacio transformado (ecuación 3.63).

$$< x_1, x_2 >= |x_1||x_2| cos(\alpha) \tag{3.61}$$

$$< x_1, x_1 >= |x_1||x_1| = |x_1|^2 \rightarrow |x_1| = \sqrt{< x_1, x_1 >} \tag{3.62}$$

$$d(x_1, x_2) = |x_1 - x_2| = \sqrt{< x_1 - x_2, x_1 - x_2 >} \tag{3.63}$$

Los SVMs en espacios transformados de los atributos toman como punto de partida la ecuación 3.60, pero utilizan una función kernel, K, para obtener los términos de la matriz de Gram: $G_{ij} = K(x_i, x_j)$. La función K permite obtener la distancia entre x_i y x_j en un espacio transformado que no es necesario obtener explícitamente. Las condiciones que debe tener K para generar matrices de Gram vienen dadas por la *condición de Mercer*, que establece que K debe ser simétrica y generar matrices semi-definidas positivas [192]. Además, Chang y Ling demostraron que si un kernel genera matrices de Gram definidas positivas (y no solamente semi-definidas positivas), los SVMs con dicho tipo de kernel pueden definir fronteras de una complejidad arbitraria [45]. Esto conlleva a que un kernel que genere matrices de Gram definidas positivas puede clasificar correctamente todos los datos de entrenamiento en un problema de clasificación binaria. En el presente trabajo se han empleado tres tipos de kernels con esta capacidad de generar fronteras de decisión de complejidad arbitraria. En primer lugar, el kernel gaussiano, un tipo de SVM muy popular y el primero para el que se probó esta capacidad (ecuación 3.64, donde σ es un parámetro del kernel) [104]. Además, también se han usado dos kernels con infinito número de hipótesis, en el contexto de clasificación binaria [120].

$$K(x_i, x_j) = e^{\frac{||x_i - x_j||^2}{2\sigma^2}} \tag{3.64}$$

Para entender qué significa que haya infinitas hipótesis se puede tomar como punto de partida una clasificación como la mostrada en 3.65, en base a la ponderación con unos pesos w_i de una serie de predictores h_i, que en este contexto llamaremos *hipótesis*. Las dos clases se codifican con $y = 1$ o $y = -1$, lo que permite el uso de la función signo. Hay un parámetro T, que es el número total de hipótesis, y que es limitado porque la forma de entrenar el clasificador se basa en un proceso iterativo, con un número de iteraciones limitadas. El método ideado por Lin y Li permite que T sea infinito [120].

$$g(x) = \text{signo}\left(\sum_{i=1}^{T} w_i h_i(x)\right) \tag{3.65}$$

Para que un Kernel contenga infinitas hipótesis, debe abarcar todas los posibles valores de las hipótesis, h_t. Esto se consigue formulando las hipótesis en función de un parámetro, α, y definiendo el kernel como una integral que recorre todos los posibles valores del parámetro. Así, llamando C al rango de posibles valores del parámetro, el espacio de hipótesis, H, es el conjunto de todas las hipótesis parametrizadas por α, $H = h_\alpha : \alpha \in C$. Cada atributo en el espacio original, x, se pasa al espacio transformado, ϕ_x, multiplicando a la hipótesis por una constante, r, que tiene que ser ajustada para que la definición del Kernel sea válida, como se muestra en 3.66. Es decir, r debe ser ajustada para que la integral exista para todos los valores de los atributos y que devuelva un valor mayor o igual a cero. Tomando la transformación de los atributos indicada, la definición del Kernel que permite construir ensembles con infinito número de hipótesis viene dada por la ecuación 3.67.

$$\phi_x(\alpha) = r(\alpha) h_\alpha(x) \tag{3.66}$$

$$K_{H,r}(x,x') = \int_C \phi_x(\alpha)\phi_{x'}(\alpha) d\alpha \tag{3.67}$$

En [120], se definen dos parametrizaciones de hipótesis que permiten construir un ensemble con infinito número de hipótesis: la parametrización de un árbol de decisión de un sólo nivel (*decision stump*, en la bibliografía inglesa) y la parametrización de un perceptrón. Para un problema con D atributos, el árbol de decisión de un sólo nivel clasifica a la clase q teniendo en cuenta únicamente al atributo d-ésimo, a través de un umbral, α, como se indica en la ecuación 3.68. Por otro lado, para el caso de un perceptrón la salida es la mostrada en la ecuación 3.69, consistente en aplicar la función signo de una combinación lineal de los atributos según unos pesos, θ, y con un umbral de separación entre clases, α. En ambos cosas se supone que los valores de x están acotados, por un hiper-rectángulo con un límite inferior L y superior R para cada dimensión en el caso del árbol de decisión de un sólo nivel, $x \subseteq [L_1,R_1] \times [L_2,R_2] \times ... \times [L_D,R_D]$, y por una bola d-dimensional de radio R para el caso del perceptrón, $x \subseteq \mathbb{B}(R)$.

$$s_{q,d,\alpha}(x) = q\,\text{signo}\left((x)_d - \alpha\right),\, q \in [-1,+1],\, d \in 1,...,D,\, \alpha_d \in [L_d, R_d] \tag{3.68}$$

$$p_{\theta,\alpha}(x) = \text{signo}(\theta^T x - \alpha), \theta \in \mathbb{R}^D,\ ||\theta||_2 = 1,\ \alpha \in [-R,+R] \tag{3.69}$$

Introduciendo las parametrizaciones 3.68 y 3.69 en 3.67 se obtienen las expresiones de los kernels con infinitos stumps (*kernel stump*, ecuación 3.70, obtenida para $r(q,d,\alpha) = \frac{1}{2}$) e infinitos perceptrones (*kernel perceptron*, ecuación 3.71).

$$K_S(x,x') = \Delta_S - ||x - x'||_1,\ \Delta_S = \frac{1}{2}\sum_{d=1}^{D}(L_d - R_d) \tag{3.70}$$

$$\begin{cases} K_P(x,x') = \Delta_P - ||x-x'||_2, \\ \Delta_P = R\left[\int_{||\theta||_2=1} d\theta\right]\left[\int_{||\theta||_2=1} |cos\left(\text{Ángulo}(\theta, e_1)\right)| d\theta\right]^{-1} \\ e_1 = (1,0,0,...,0)^T \end{cases} \tag{3.71}$$

Los problemas de optimización vistos anteriormente están asociados a una clasificación binaria. El problema multiclase se resuelve a partir de una descomposición en varios binarios, como se explica en el artículo "*Identifying maximum imbalance in datasets for fault diagnosis of gearboxes*" (sección 5.2). En 4.3 se profundiza en la idea de margen, y en cómo usar la idea de margen para problemas de regresión.

Redes neuronales

Las redes neuronales surgen del intento de emular el comportamiento de un cerebro humano, dotando a un sistema artificial de la capacidad de razonamiento [200]. Para ello, los modelos matemáticos recogen dos aspectos obtenidos en estudios sobre la naturaleza del cerebro humano. En primer lugar, su estructura a base de elementos muy simples, las neuronas, trabajando conjuntamente para resolver problemas muy complejos en lo que se conoce como *procesamiento en paralelo*. En segundo lugar, la direccionalidad de las *sinapsis* o transmisión de información entre neuronas, que siempre se produce en el mismo sentido. Las redes neuronales artificiales se basan en una simplificación del modelo biológico de una neurona que tiene las dos propiedades anteriores, por lo que en ocasiones

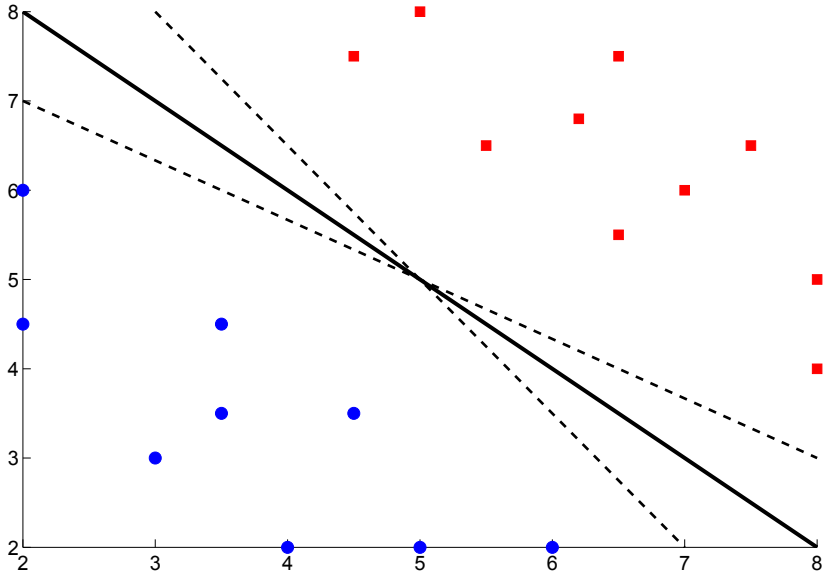


Figura 3.10: Idea de maximización del margen en el clasificador SVM

también reciben el nombre de sistemas de procesamiento en paralelo (*Parallel Distributed Processing systems (PDPs)*) o sistemas conexionistas (*Connectionist Systems*) [132]. La primera neurona artificial fue diseñada por McCulloch-Pitts en 1943 [182], constituyendo una función *booleana* (con dos estados posibles de salida, activa o desactivada) de una transformación lineal de las entradas a la neurona. Dicho modelo se muestra en 3.72, siendo f la función de activación, x_i cada una de las N entradas a la neurona, θ el umbral de activación o *bias* e y la salida de la neurona.

$$\begin{cases} y = f(\sum_{i=1}^{N} w_i x_i - \theta) \\ f(u) = 1 \text{ si } u \geq 0 \\ f(u) = 0 \text{ si } u < 0 \end{cases} \tag{3.72}$$

Existen diversos tipos de configuraciones de redes neuronales, habiéndose usado en el presente estudio el más popular de ellos, conocido como *perceptrón multicapa* (*multi-layer perceptron*, MLP, en la bibliografía inglesa). Por ello, la descripción presentada en este apartado se refiere principalmente a redes MLP. Sin embargo, también se comenta brevemente la estructura de un tipo de red neuronal a la que se hace referencia al hablar

de las máquinas de vectores soporte: las *redes con función base radial* (*radial basis function network*, RBFN, en la bibliografía inglesa). Las redes MLP usadas en este estudio tienen tres capas, una con las entradas a la red (atributos del conjunto de datos), una capa oculta, y una capa de salida donde se asigna la predicción a cada instancia. En primer, se calcula la salida de la capa oculta, y_{oculta}, a partir de los atributos, y después, se obtiene la salida, y_{salida}, tomando y_{oculta} como entrada en la capa de salida, como se muestra en la ecuación 3.73, donde W_1 y W_2 son las matrices de pesos de la capa oculta y de salida, f_{red} la función de activación de la capa oculta, y f_{salida} la función de activación de la capa de salida. La primera dimensión de la matriz W_1 es un parámetro del sistema de predicción, el número de neuronas de la capa oculta (n_o), mientras que el resto de dimensiones vienen dadas por el número de atributos de entrada, m, y por las dimensiones de la salida, n_s. ($W_1 : n_o \times (m+1)$, $W_2 : n_s \times (n_o+1)$). A los atributos iniciales del problema, de dimensión m, se les añade un atributo adicional de valor constante, $x_0 = 1$, como se vio con la regresión lineal. De esta forma los umbrales de activación no aparecen explícitamente en la formulación y los vectores de atributos, x, pasan a tener dimensión $(m+1) \times 1$. También en la capa de salida se añade una dimensión a y_{oculta} para prescindir del umbral de activación en la formulación. Por ello, se denota y'_{oculta} al vector resultante de añadir a y_{oculta} una componente de valor 1. Respecto de las dimensiones de la salida, para problemas de regresión se toma una salida unidimensional $n_s = 1$, mientras que para clasificación se predicen las probabilidades de las K clases del problema, $n_s = K$.

$$\begin{aligned} y_{oculta} &= f_{red}(W_1 x) \\ y_{salida} &= f_{salida}(W_2 y'_{oculta}) \end{aligned} \tag{3.73}$$

En el caso de las redes RBFN la salida se obtiene en una única fase, mediante una media ponderada de las salidas de la capa oculta. En cada una de las unidades de la capa oculta se almacena un patrón, de modo que la predicción se basa en la similitud observada respecto con los patrones almacenados. Así, cada neurona de la capa oculta tiene una función de activación, Φ, que proporciona la similitud con el patrón almacenado, x_q. La similitud entre los atributos de la instancia a predecir, x, y el patrón almacenado, x_q, viene dada por la norma de la diferencia, $||x - x_q||$, y el valor de la clase predicho por la red RBFN, $f(x)$, se expresa a través de la ecuación 3.74, donde w_q es el peso de la neurona q-ésima de la capa oculta. Por tanto, $f(x)$, es un desarrollo en serie de las funciones base,

Φ. El nombre que reciben las funciones base de *radiales* viene dado por que la norma $||x-x_q||$ puede verse como el radio, r, que separa al patrón almacenado en cada neurona y los atributos de la instancia cuya clase se quiere predecir. La función radial más frecuentemente usada es la gaussiana, dada por la expresión 3.75, done σ es el parámetro de anchura.

$$f(x) = \sum_{q=1}^{n_o} w_q \Phi\left(||x-x_q||\right) \tag{3.74}$$

$$\Phi(r) = e^{-\frac{r^2}{2\sigma^2}} \tag{3.75}$$

Volviendo al tipo de redes neuronales usadas en el presente estudio, las redes MLP de tres capas, la explicación de las posibles funciones de activación tanto de la capa oculta como de la capa de salida enlaza con el desarrollo que se hizo de la regresión logística, ya que también para las redes neuronales se usan *Logit* y *Softmax*, si bien en este caso se emplea una tercera función no lineal, la tangente hiperbólica, *Tanh*, definida en 3.76. Se puede comprobar que *Tanh* es una transformación de *Logit*, sumando 1 a *Tanh* y después multiplicando por e^x a numerador y denominador, como se muestra en 3.77. Mientras que la salida de *Logit* está en el rango [0,1], *Tanh* recorre los mismos valores en una escala el doble de grande y con los valores centrados en una unidad menos. Además, los valores de *Tanh* tienen el doble de separación en el eje x que los de *Logit*.

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.76}$$

$$\begin{cases} Tanh(x)+1 = \frac{e^x-e^{-x}}{e^x+e^{-x}} + \frac{e^x+e^{-x}}{e^x+e^{-x}} = 2\frac{e^x}{e^x+e^{-x}} = 2\frac{e^{2x}}{e^{2x}+1} = 2Logit(2x) \\ Tanh(x) = 2Logit(2x) - 1 \end{cases} \tag{3.77}$$

Se tienen diferentes configuraciones posibles en lo que se refiere al número de salidas y a la función de activación de la capa de salida, dependiendo del tipo de problema considerado. Las configuraciones más habituales son:

- Para problemas de clasificación binaria, una única salida con la función de activación *Logit*.

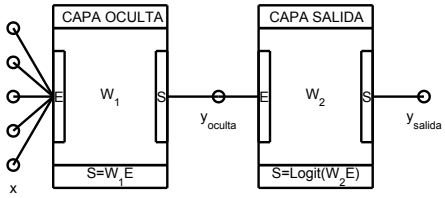
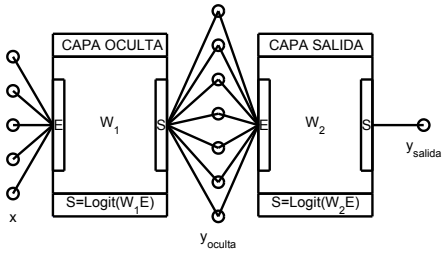
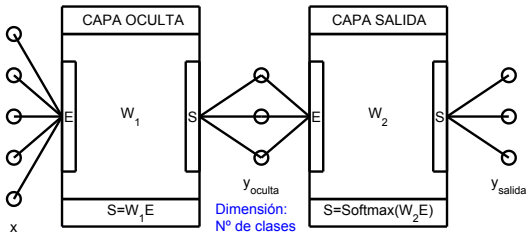
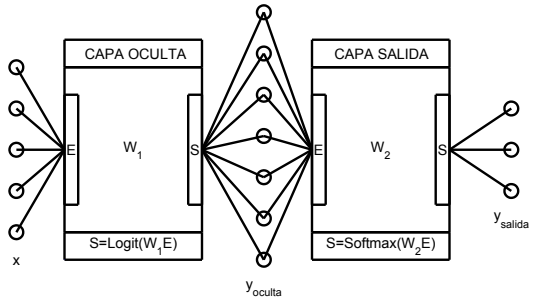
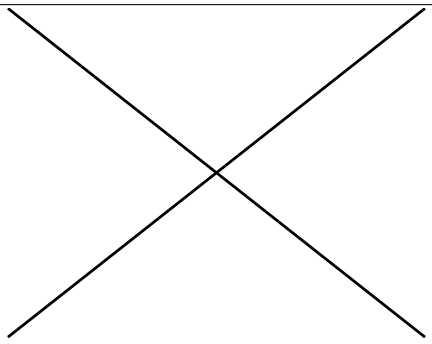
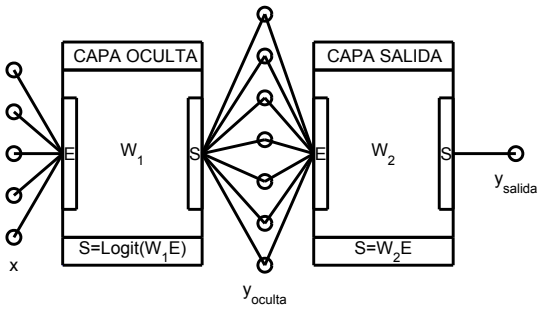
- Para problemas de clasificación multiclase, tantas salidas como clases, con la función de activación *Softmax*.
- Para problemas de regresión, hay una única salida para la que no se aplica ninguna transformación adicional en base a una función, sino que se obtiene mediante una combinación lineal de las entradas a la capa.

Respecto de la función de activación de la capa oculta, aunque en diferentes estudios teóricos se ha contemplado tanto el uso de *Logit* como de la tangente hiperbólica, *Tanh*, en la implementación práctica en los diferentes paquetes de software se prefiere *Tanh*, debido a que tiende a converger más rápidamente [117]. La razón de la mejor convergencia de *Tanh* es su simetría respecto del origen, lo que hace que sea más probable que produzca salidas de media cero.

La regresión logística, tanto en su versión de clasificador binario como multiclase, puede verse como un caso particular de redes neuronales en los que la función de activación de la capa oculta es lineal y el número de neuronas de la capa oculta está prefijado [15]. Así, para el caso binario se tiene una única neurona en la capa oculta, mientras que para el caso multiclase se tienen tantas neuronas como clases (la dimensión de la salida de la capa oculta es igual, por tanto, a la dimensión de la salida de la capa de salida). En el Cuadro 3.5 se muestran las configuraciones de las redes neuronales y la regresión logística en los casos de clasificación binaria y multiclase, así como la configuración de las redes neuronales en caso de regresión. Las redes neuronales pueden verse como una generalización de la regresión logística, añadiendo una transformación no lineal *Logit* (o su equivalente *Tanh*) al modelo lineal que se tenía para la oculta en el caso de la regresión logística, y con un número variable de neuronas en la capa oculta.

El hecho de que las funciones de activación usadas por la capa oculta de una red neuronal MLP sean *Logit* o *Tanh* no viene dado por su relación con la regresión logística, sino a que tomando funciones sigmoideas en la capa oculta las redes neuronales pueden aproximar a cualquier función continua y medible, como demostró el trabajo de Hornik, Stinchcombe y White [88]. *Logit* y *Tanh* son dos tipos de funciones sigmoideas que resultan ventajosas en el entrenamiento de una red neuronal, como se verá más adelante. Se dice que una función real es sigmoidea cuando tiene forma de "S" y salida en el rango (-1,+1). Matemáticamente, una función σ, real con salida en (-1,+1), $\sigma : \mathbb{R} \to (-1, +1)$, es sigmoidea si cumple cuatro condiciones [128]:

Cuadro 3.5: Configuraciones de redes neuronales y regresión logística

	Logística	Redes
Clas. Binaria		
Clas. Mutliclase		
Regresión		

- Es continuamente diferenciable (las derivadas parciales de cualquier orden son continuas).
- Es impar, $\sigma(-x) = -\sigma(x)$.
- Tiene asíntotas horizontales en -1 y +1, $\lim_{x\to-\infty}\sigma(x) = -1$, $\lim_{x\to+\infty}\sigma(x) = +1$.
- $\sigma(x)/x$ es completamente convexa (tiene derivadas de cualquier orden, y para la derivada k-ésima de orden par, $f^{(2k)}$, se cumple que $(-1)^k f^{(2k)}(x) \geq 0$).

El entrenamiento de las redes neuronales MLP de tres capas tiene dos fases:

- Fase de *alimentación hacia adelante* (*feed-fordward*, en la bibliografía inglesa). Se calcula el error entre la salida predicha por la red y el valor real de la clase. El sentido del flujo de información en esta etapa es desde la entrada hacia la salida. Es decir, primero se calcula la salida de la capa oculta y después la salida de la capa de salida.
- Fase de *retropropagación* (*backpropagation*, en la bibliografía inglesa). Se actualizan los pesos de la red a partir del error cometido. El sentido del flujo de información en esta etapa es desde la salida hacia la entrada. Es decir, primero se actualizan los pesos de la capa de salida y después los de la capa oculta. Es esta segunda fase la que suele dar el nombre a la técnica de entrenamiento, que se conoce genéricamente como *backpropagation*.

En la gráfica 3.11 se muestra el esquema del entrenamiento de las redes neuronales, *backpropagation*, con las fases con sentido inverso de flujo de información.

Los pesos se actualizan mediante técnicas de gradiente descendente 3.78. El error no depende directamente de los pesos w, sino de las salidas de las capas, y_{capa}. Por ello, se introduce en 3.79 la dependencia de y_{capa} a través de la regla de la cadena, que determina la derivada de la composición de funciones, $g(f(x))$, a través del producto $g'(f(x))f'(x)$. Entonces, aparece en el cálculo la derivada de la salida de la capa respecto de los pesos, $\frac{\partial y_{capa}}{\partial w}$. Dicho término tiene una expresión variable, dependiendo de que tipo de función de activación tenga la capa. Por ello, es conveniente aplicar la regla de la cadena nuevamente, definiendo la activación de la capa como net_{capa}. De este modo se separa la parte lineal, net_{capa}, que no depende de la configuración de la red. En la ecuación 3.79 se muestra

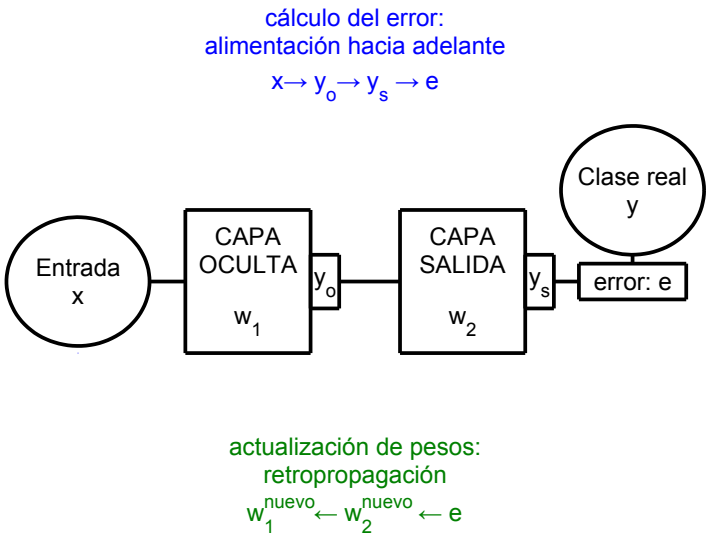


Figura 3.11: Esquema del entrenamiento *Backpropagation*

la descomposición del gradiente del error respecto al peso en los tres términos citados. Denotando por φ a la función de activación de la capa y teniendo en cuenta la linealidad de net_{capa} respecto a las entradas de la capa, el gradiente del error se puede expresar de forma resumida como se indica en 3.80.

$$\Delta w = -\eta \frac{\partial E}{\partial w} \tag{3.78}$$

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial y_{capa}} \frac{\partial y_{capa}}{\partial w}, \ \frac{\partial y_{capa}}{\partial w} = \frac{\partial y_{capa}}{\partial net_{capa}} \frac{\partial net_{capa}}{\partial w} \tag{3.79}$$

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial y_{capa}} \varphi'_{capa}(net_{capa}) x_{capa} \tag{3.80}$$

Para el cálculo de la derivada del error respecto de la salida de la capa, hay que tener en cuenta que el error está directamente relacionado con la salida de la capa de salida, pero no con la salida de la capa oculta. Por ello, se puede derivar directamente para la capa de salida, pero para obtener el valor en la capa oculta hay que partir del valor de la capa de salida. Así, en una única componente escalar de la capa oculta influyen todas las

componentes de la capa de salida, ya que todas están conectadas. Por ello, la expresión para la derivada de la capa oculta depende de la suma para todos las neuronas de la capa de salida. En la ecuación 3.81 se muestran las expresiones de la derivada de las salidas de la capa de salida y de la capa oculta.

$$\begin{cases} \frac{\partial E}{\partial y_s} = E'(y_s) \\ \frac{\partial E}{\partial y_{o_j}} = \sum_{i=1}^{n_s} \frac{\partial E}{\partial y_s}\frac{\partial y_s}{\partial y_o} = \sum_{i=1}^{n_s} \frac{\partial E}{\partial y_s}\frac{\partial y_s}{\partial net_s}\frac{\partial net_s}{\partial y_o} = \sum_{i=1}^{n_s} E'(y_s)\varphi_s'(net_s)\{W_2\}_{ij} \end{cases} \tag{3.81}$$

Sustituyendo 3.81 en 3.80 se tiene la regla de actualización de los pesos, ecuación 3.82. El término $E'(y_s)$ depende de cuál sea la definición de la función objetivo a minimizar, existiendo varias posibilidades tanto para regresión como para clasificación. También existen varias posibilidades para φ, siendo las más comunes *Logit* y *Tanh*, cuyas derivadas se muestran en 3.83. El cálculo de las derivadas para ambas funciones resulta ventajoso respecto de otras funciones de activación porque se pueden obtener a partir del propio valor de la función. Por tanto, si se almacena el resultado al calcular la salida no es necesario volver a computar la operación para calcular la derivada en la actualización de pesos.

$$\begin{cases} \Delta w_2 = -\eta E'(y_s)\varphi_s'(net_s)y_o \\ \{\Delta w_1\}_j = \eta \left[\sum_{i=1}^{n_s} \left(E'(y_s)\varphi_s'(net_s)\{W_2\}_{ij}\right)\right] \varphi_o'(net_o)x \end{cases} \tag{3.82}$$

$$\begin{cases} \varphi(z) = Logit(z) \rightarrow \varphi'(z) = \frac{e^{-z}}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}}\frac{e^{-z}}{1+e^{-z}} = \varphi(z)\left(1-\varphi(z)\right) \\ \varphi(z) = Tanh(z) \rightarrow \varphi'(z) = \frac{\left(e^z+e^{-z}\right)^2-\left(e^z-e^{-z}\right)^2}{(e^z+e^{-z})^2} = 1-\left[\frac{e^z-e^{-z}}{e^z+e^{-z}}\right]^2 = 1-\varphi^2(z) \end{cases} \tag{3.83}$$

Hay tres tipos de entrenamiento *backpropagation*:

- Si el cálculo del error tiene en cuenta todas las instancias, se habla de *entrenamiento por lotes* (*batch training*, en la bibliografía inglesa). No hay actualización de pesos hasta que se han calculado las clases predichas para todas las instancias. En este contexto se habla de *épocas* (*epochs*, en la bibliografía inglesa), el número total de actualizaciones de pesos, equiparable al número de *iteraciones* para otros tipos de entrenamiento [19].

- Si el cálculo del error tiene en cuenta sólo a una instancia, habiendo actualización

de pesos con cada clase predicha, existen dos opciones.

- En el entrenamiento *estocástico* (*stochastic learning*, en la bibliografía inglesa), se escogen las instancias de forma aleatoria hasta alcanzar el criterio de parada. De este modo, cada instancia podría ser utilizada varias veces o ninguna.
- En el entrenamiento *en línea* (*on-line learning*, en la bibliografía inglesa), cada instancia se utiliza una única vez, no habiendo almacenamiento de datos. Este tipo de entrenamiento se emplea en contextos diferentes al del presente estudio, ya que es útil cuando no existe un conjunto fijo de datos de entrenamiento, sino que éstos van cambiando con el tiempo.

Respecto de la función de error, lo más común es minimizar el error cuadrático para problemas de regresión y la entropía cruzada, como en la regresión logística, para problemas de clasificación [75].

El algoritmo de entrenamiento *backpropagation* puede presentar dos problemas en su implementación práctica [98]:

1. La convergencia del método puede ser muy lenta si la función de error encuentra mínimos en entornos con forma de "valle" (con poca pendiente, con valores pequeños del gradiente).

2. Las redes neuronales pueden sobreajustar, llegando a aprender el ruido en los datos. Esto en ocasiones se produce cuando el valor de algunos pesos se hace muy grande.

Existen en la literatura diferentes soluciones para los dos problemas anteriores, entre las que destacamos una para cada problema. El problema de la lentitud en la convergencia puede afrontarse añadiendo un *momentum*, α, que reduce las oscilaciones del proceso de búsqueda según la dirección de gradiente en presencia de valles, reorientando la dirección de búsqueda hacia el centro del valle [156]. Así, a la expresión de la variación de un peso, Δw, para la iteración i-ésima, se le añade un término que depende de la variación en la iteración anterior, $\Delta w(i-1)$, y que se le suma al que se tenía en el entrenamiento backpropagation, como se muestra en la ecuación 3.84.

$$\Delta w(i) = -\eta \frac{\partial E}{\partial w} + \alpha \Delta w(i-1) \tag{3.84}$$

Por otro lado, para evitar el sobreajuste es posible aplicar técnicas de *degradación de pesos* (*weight decay*, en la bibliografía inglesa). Para ello, la actualización de los pesos en cada iteración no parte directamente del valor anterior que tenían dichos pesos, sino que son multiplicados previamente por un factor, λ, como se indica en la ecuación 3.85 [77]. El valor de λ se fija ligeramente menor que la unidad (por ejemplo, 0.9), con lo que todos los pesos se reducen algo en cada iteración respecto al valor que tendrían con la regla de *backpropagation*, pero penalizando más a los pesos más grandes.

$$w(i) = \lambda w(i-1) + \Delta w(i) \tag{3.85}$$

3.5.4. Predicción basada en instancias

A diferencia del resto de técnicas de predicción vistas, en las que se construye durante la fase de entrenamiento un modelo que describe la relación entre los atributos de entrada y los atributos a predecir, en este tipo de predicción no se realiza ningún cómputo durante el entrenamiento, sino que simplemente se almacenan las instancias, posponiendo todo el cálculo para la fase de predicción [2]. El hecho de no tener entrenamiento, hace que también se les conozca como técnicas de "aprendizaje perezoso" (*lazy learning*) o "basadas en memorizar" (*memory-based*). Para asignar la clase a cada instancia se analizan los valores de las instancias con atributos más parecidos de entre todas las guardadas en el entrenamiento. El algoritmo más representativo de este tipo de técnicas de predicción es el de los k-vecinos [1].

En el caso de que esta técnica se use para clasificación, a cada instancia de test se le asigna la clase de los vecinos más cercanos a la misma, almacenados en el entrenamiento, como se muestra en la Figura 3.12. En dicha figura se toma como ejemplo la fiabilidad de una máquina, representando con puntos rojos los fallos, con cuadrados azules funcionamiento correcto y el punto verde la instancia a clasificar. Tomando un único vecino, dentro del círculo más pequeño en la figura, la instancia sería clasificada como correcta, pero tomando tres vecinos sería clasificada como fallo.

Si se tienen atributos numéricos, previamente se les aplica una transformación que hace que varíen en el rango $[0,1]$, lo que se conoce como normalización. Con esto se consigue que todos los atributos tengan una influencia similar en la distancia total, no importando las escalas usadas. Dado un vector $\bar{x}$, con componentes x_i, su expresión normalizada, x_i^{norm},

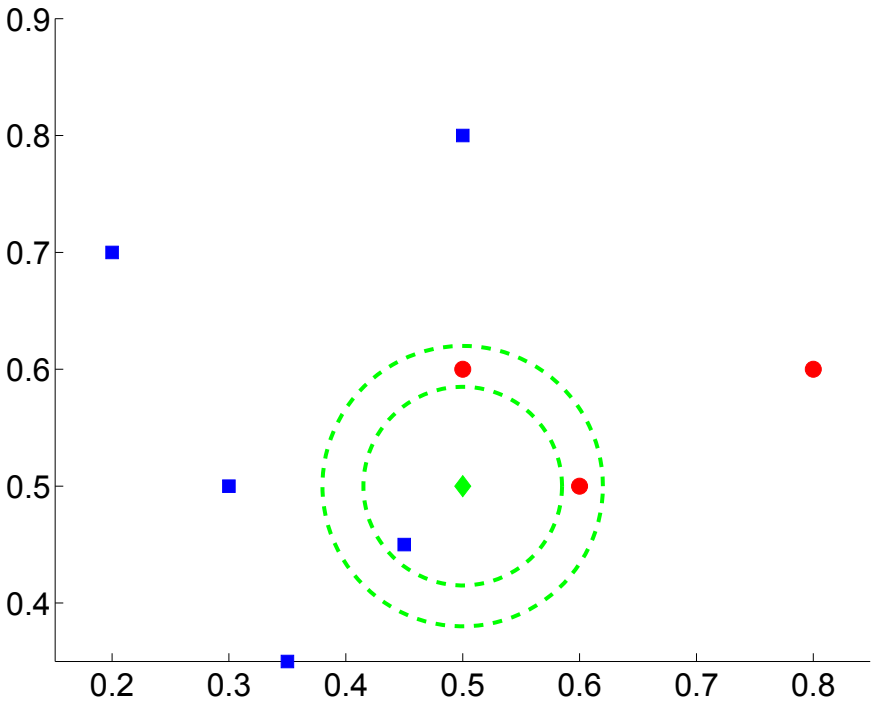


Figura 3.12: Funcionamiento del clasificador de los k vecinos

viene dada por la expresión 3.86.

$$x_i^{norm} = \frac{x_i - min(x_i)}{max(x_i) - min(x_i)} \tag{3.86}$$

Una vez que los atributos están normalizados, la distancia (d) entre dos instancias, $\overline{x}$ e $\overline{y}$ viene dada generalmente por la norma euclídea de su diferencia (3.87).

$$d = |\overline{x} - \overline{y}| = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2} \tag{3.87}$$

Para el caso de atributos nominales, una forma común de medir la distancia es a través del número de coincidencias (c) y el número total de variables (n), a partir de la expresión 3.88.

$$d = \frac{n - c}{n} \tag{3.88}$$

Para el caso de la regresión también se construye la predicción a partir de los k vecinos más cercanos, pero en este caso a partir del promedio de los valores de la clase para dichas instancias, como muestra la ecuación 3.89, donde y_e es el valor estimado e y_i son los valores de la clase para cada uno de los vecinos.

$$y_e = \frac{1}{k} \sum_{i=1}^{k} y_i \tag{3.89}$$

Este tipo de aprendizaje puede sufrir lo que se conoce como *maldición de la dimensionalidad* (*curse of dimensionality*, en la bibliografía inglesa) [92]. Este problema surge al analizar ciertos procesos cuyas propiedades son conocidas en el espacio físico (en 3 dimensiones), pero que presentan diferente comportamiento con múltiples dimensiones. Tal es el caso de la distancia euclídea con múltiples dimensiones, donde las distancias entre instancias tienden a ser cada vez más grandes. Así, podríamos imaginar un problema con muchas dimensiones pero que dos de ellas fuesen más relevantes que las demás, que sólo contendrían información complementaria. Entonces, dos instancias con valores idénticos de los atributos más significativos podrían estar muy distantes. También podría ser problemático tener dos atributos que aportasen información similar. En tal caso dicho atributo se tendría en cuenta dos veces a la hora de calcular la distancia, haciendo que la influencia de la información común a esos dos atributos fuera excesivamente decisiva a la hora de

decidir qué instancias fueran vecinas.

3.5.5. Ensembles

Los *ensembles* (o *multiclasificadores* en problemas de clasificación, y por extensión *multiregresores* en problemas de regresión) construyen un sistema de predicción a partir de la combinación de varios predictores, conocidos como predictores base [172]. De esta forma se busca que la predicción final sea más precisa que la asignada por cada predictor individual. Existen varias formas de combinar los predictores base, ya que todos ellos pueden tener el mismo peso en la decisión final o puede utilizarse un voto ponderado en función de su precisión sobre los datos de entrenamiento.

Los ensembles pueden entenderse de manera intuitiva haciendo un símil con el proceso de toma de decisiones que también seguimos los humanos en el día a día [144]. La idea de este método de predicción es "consultar a varios expertos antes de decidir". Por ejemplo, preguntar a varios doctores antes de tomar la decisión de operarse.

Rokach [157] distingue 3 estructuras diferentes a lo hora de combinar las predicciones de los predictores base para obtener la predicción final del ensemble:

1. *Fusión* de las predicciones base mediante un sistema de voto, bien sea ponderado o mayoritario.

2. *Selección* de la salida de un único predictor base. Para ello, a menudo el conjunto de entrenamiento es dividido en particiones o bien de las muestras o bien de los atributos.

3. *Meta-aprendizaje*, en el cual el entrenamiento tiene dos etapas. Inicialmente se entrenan los predictores base, y, posteriormente, se construye un predictor en un nivel jerárquico superior a partir de las salidas de los predictores base. Esta estructura es más adecuada para problemas en los que hay ciertos grupos de instancias diferentes, en los que algunos predictores base pueden tener mayor o menor precisión.

En el presente Trabajo de Investigación se han usado ensembles que se pueden encuadrar dentro del primer tipo. No se han considerado adecuados ensembles ni del segundo ni del tercer tipo, ya que a priori se espera homogeneidad de las instancias en los conjuntos de los problemas industriales analizados.

También de forma intuitiva, el trabajo de Brown [32] analiza las condiciones para que un pronóstico basado en múltiples predictores sea mejor que una predicción con un único predictor, empleando también una analogía con la toma de decisiones a través de un comité de expertos. Sólo sería beneficioso tal proceso de decisión si los expertos no se equivocan en los mismos puntos, en otras palabras, es necesario que sus opiniones no sean idénticas. El mismo principio es aplicable para combinación de técnicas de reconocimiento de patrones: se busca que cada predicción tenga un grado de precisión aceptable, pero existiendo cierta *diversidad* entre las mismas. Tomando como referencia este concepto de *diversidad*, Brown clasifica a los ensembles en dos categorías según la forma de promover la diversidad entre predictores base, aquellos que la promueven de forma *explícita* e *implícita*. En los ensembles con diversidad implícita se espera que los predictores sean diferentes al ser entrenados con conjuntos de datos distintos, pero no establece ninguna medida para asegurarse de que realmente existan dichas diferencias. En los ensembles con diversidad explícita, sin embargo, sí aparece en la formulación un elemento que fuerza las diferencias entre predictores base, como un sistema de repesado de las instancias.

Se ha usado la taxonomía empleada por Brown porque por su simpleza facilita la comprensión de los ensembles, pero pueden encontrarse múltiples clasificaciones de los ensembles en la literatura, de acuerdo a un único criterio o a varios simultáneamente [155]. Sin embargo, centrarse en la idea de diversidad como en la taxonomía de Brown permite también analizar qué tipo de predictores base son adecuados para los ensembles. En la mayoría de los ensembles es beneficioso tomar predictores base con técnicas de entrenamiento inestables, como los árboles de decisión [32]. Pero esto no quiere decir que necesariamente haya que utilizar técnicas de entrenamiento *inestables* (i.e., que los modelos generados cambien considerablemente al introducir pequeños cambios en el conjunto de entrenamiento), ya que algunos ensembles pueden forzar la diversidad incluso con técnicas de predicción muy estables, como los SVMs. Esta limitación es mayor en los ensembles que promueven la diversidad de forma implícita, y especialmente para el caso de Bagging. Sin embargo, también existen trabajos en los que ensembles de predictores con técnicas de entrenamiento estables como los SVMs han dado buenos resultados [189]. En el presente Trabajo de Investigación se han usado mayoritariamente árboles de decisión como predictores base, si bien para el estudio de la predicción de errores en aerogeneradores también se tomaron como predictores base el método de los k-vecinos y el

Bayesiano Ingenuo, como puede consultarse en el artículo "*Wind turbines fault diagnosis using ensemble classifiers*" (sección 5.2).

Se han utilizado seis tipos de ensembles, tres en los que se fuerza la diversidad de forma implícita (Bagging, Random Subspaces y Rotation Forest), uno en el que ésta se fuerza de forma explícita (Boosting), y dos métodos en los que se fuerza la diversidad de ambas formas (Iterated Bagging y Regresión Aditiva), ya que son ensembles híbridos entre Bagging y Boosting.

Promoción interna de la diversidad

Una forma de lograr la diversidad entre los predictores base es entrenarlos en subconjuntos diferentes obtenidos a partir del conjunto de entrenamiento inicial. Tres son los tipos de ensembles usados en el presente Trabajo de Investigación que siguen esta estrategia:

- En Bagging, los subconjuntos generados tienen menos instancias que los datos iniciales, y se obtienen por muestreo con reemplazo (cada instancia puede ser seleccionada cero o varias veces en el mismo conjunto).

- En Random Subspaces, cada subconjunto se forma tomando una parte aleatoria de los atributos del conjunto inicial.

- En Rotation Forest, cada subconjunto se obtiene a partir de una proyección de los datos mediante una matriz de rotación basada en un *Análisis de Componentes Principales* (*Principal Component Analysis*, PCA, en la bibliografía inglesa). Para la generación de dicha matriz de rotación se sigue una estrategia más compleja que en los dos ensembles anteriores. Inicialmente, se dividen los atributos en varios grupos y se forman unos subconjuntos auxiliares a partir de dicha división de los atributos. Posteriormente, para cada subconjunto auxiliar se obtiene una matriz de proyección PCA pero no utilizando todos los datos, ya que previamente se eliminan aleatoriamente parte de las clases y se toma una muestra de las instancias, como en Bagging. La matriz de rotación resultante se obtiene a partir de las matrices de proyección de cada uno de los grupos de atributos.

Promoción externa de la diversidad

Como representante de ensembles en lo que se promociona de forma externa la diversidad entre predictores base, se han usado varias técnicas de predicción que tienen su base en el algoritmo AdaBoost (*Adaptive Boosting*). Dicho algoritmo es una implementación de *Boosting*, palabra inglesa formada a partir del verbo *boost*, que significa *estimular, impulsar*. Boosting se fundamenta en la teoría matemática del *aprendizaje aproximadamente correcto* (*probably approximately correct learning, PAC learning*, en la bibliografía inglesa), propuesta por Leslie Valiant en 1984 [190], y a partir de la cual se planteo la forma de combinar varias predicciones *débiles* con acierto ligeramente mayor a una predicción aleatoria, hasta obtener una predicción *fuerte* [103].

En el caso de la implementación de Boosting mediante el algoritmo AdaBoost, se utiliza una técnica de aprendizaje secuencial para forzar la diversidad entre los predictores base, de forma que cada predictor se centra más en aquellas instancias en las que el predictor anterior obtuvo peores resultados. Una posible solución se consigue mediante unos pesos que se asignan a las instancias de entrenamiento, y que inicialmente son iguales, pero que se incrementan con el error de predicción. Esta solución se llama *repesado*, y es que la que se va a detallar a continuación por haber sido usada en el presente Trabajo de Investigación, aunque también existe otra solución conocida como *remuestreo*.

El predictor final se construye como una media ponderada de los predictores débiles, según la ecuación 3.90, en la que β_i es el peso de cada uno de los N predictores débiles, h_i, y H es la predicción final. Tras cada iteración se modifica el conjunto de entrenamiento para el siguiente h_i, multiplicando a los pesos de cada instancia, w, por un factor, $\alpha(e)$, que depende del error cometido sobre dicha instancia, e, y luego normalizando para que la suma de todos los pesos sea 1. La expresión de los pesos, β_i, es tal que minimiza una función de coste, C. En 1 se muestra un esquema del procedimiento de entrenamiento seguido por Adaboost, donde la definición del error total, E, no tiene porque coincidir con la función de coste, C, que se minimiza para calcular los pesos de cada predictor base, β_i. En la versión inicial de AdaBoost para clasificación (M1), e se calcula a partir de la función de pérdida 0/1, pero como a partir de dicha pérdida no puede encontrarse un mínimo, se usa una función de coste exponencial en su lugar. Freund encontró a través de la teoría PAC que la pérdida 0/1 estaba acotada superiormente por la pérdida exponencial, definiendo un factor de actualización $\alpha(e) = exp(e)$ [68].

$$H = \sum_{i=1}^{N} \beta_i h_i \tag{3.90}$$

Pseudocodigo 1 Modificación de los conjuntos de entrenamiento T en AdaBoost

```
Inicial: w de cada instancia iguales
Para cada iteración i de 1 a N {
    entrenar hi con T
    calcular e de hi para cada instancia de T
    calcular E total
    modificar w de cada instancia de T:  w=w x alfa(e)
    recalcular w para que su suma sea 1
}
Obtener los beta que optimicen función de coste C

Predicción final: media ponderada de los hi según coeficientes beta
```

Por tanto, AdaBoost puede ser visto también como un método iterativo de optimización de la función de coste [123]. Las funciones de coste se construyen a partir de las funciones de pérdida vistas en el Apartado 3.4.2, y que se refieren a una única instancia (el término l que se definió entonces es análogo al término e en Adaboost). Existen dos estrategias diferenciadas para la resolución de dicha optimización según se defina el factor de actualización de los pesos. Por un lado, hay variantes de Adaboost inspiradas en su primera formulación, obtenida a través de la teoría PAC, y, por otro, hay un procedimiento sistemático de resolver el problema de minimización para una función de coste genérica a través de técnicas de búsqueda de gradiente descendente. Las variantes de AdaBoost que emplean esta técnica de minimización reciben el nombre genérico de *Gradient Boosting* [158].

La minimización mediante el gradiente descendente pertenece a la familia de *métodos de descenso* para la búsqueda de mínimos de funciones. Dichos métodos consisten en un proceso iterativo en el que la función que desea minimizar, $f(x)$, va *descendiendo* de valor con cada nueva iteración. Para ello, la variable de la función, x, es modificada en cada iteración, k, de acuerdo a una dirección de búsqueda, Δx y un ancho de paso, t, según indica la ecuación 3.91 [27].

$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)} \tag{3.91}$$

Para las funciones de tipo convexo, si el método converge a un mínimo local éste es además global, debiendo formar la dirección de búsqueda un ángulo agudo con el gradiente negativo de la función para garantizar la convergencia (lo que se denomina dirección *descendente*), como se muestra en la ecuación 3.92 [27]. Es la variante del método de descenso en la que la dirección de búsqueda es el valor negativo del gradiente de la función, $\Delta x = -\nabla f(x)$, la que recibe el nombre de método del gradiente descendente.

$$\nabla f\left(x^{(k)}\right)^T \Delta x^{(k)} < 0 \tag{3.92}$$

Una vez que se ha descrito el proceso de minimización de la función de coste, se explican a continuación las diferentes posibilidades para la definición de dicha función en los dos tipos de problemas de predicción tratados: la clasificación y la regresión. Como se indicó al explicar la validación de predictores, el grado de precisión de un predictor se calcula tomando un valor agregado (tal como la media o el total) de la función de pérdida para el conjunto de test. En este caso también se usan valores agregados de las funciones de pérdida, pero tomando el propio conjunto de entrenamiento. En el presente Trabajo de Investigación se han usado tres variantes de AdaBoost: para clasificación, M1, y para regresión, R y Regresión Aditiva (este último ensemble, además del esquema de AdaBoost tiene una modificación inspirada en Bagging, como se explica más adelante). En el Cuadro 3.6 se detallan la definición de pérdida que se hace para cada instancia en las diferentes variantes, definidas a partir del valor real, y_r, y el valor estimado, y_e. El sistema de minimización del coste se basa en la definición inicial de Adaboost para M1 y R, mientras que la Regresión Iterativa utiliza una búsqueda de gradiente descendente. Con el tipo de pérdida definida para este último caso el gradiente, $\nabla(e)$, toma una expresión muy simplificada, al ser simplemente la resta del valor real y el predicho. A esta diferencia se la denomina *residuo*, concepto en el que se incide en el siguiente apartado.

No todas las funciones de pérdida son adecuadas para Gradient Boosting, ya que como se ha explicado, el método del gradiente descendente se aplica a funciones convexas tomando el valor negativo del gradiente de la función como dirección de búsqueda. Esto implica que se debe usar una función convexa y que la complejidad de su gradiente influye en la eficiencia computacional. Estas restricciones no son problemáticas para el caso de

Cuadro 3.6: Pérdidas usadas en las variantes de AdaBoost

M1	$e = 0$ si $y_e = y_r$, $e = 1$ en otro caso
R	$l = \|y_r - y_e\|$, Den valor en instancia con máximo l $e_{exponencial} = 1 - exp(l - Den)$ $e_{lineal} = l/Den$ $e_{cuadrada} = [l/Den]^2$
Regresión Aditiva	$e = 1/2\,(y_r - y_e)^2 \rightarrow \nabla(e) = y_r - y_e$

las funciones de pérdidas más habituales en regresión (como el error cuadrático o el valor absoluto), pero sí para la función de pérdida más frecuente en clasificación, la pérdida 0/1, que no es convexa. Para ello surge el concepto de "funciones sustitutas" (*surrogate loss functions*, en la bibliografía inglesa), que reciben este nombre por remplazar a la función de pérdida 0/1, pero que sí son convexas [213]. Dichas "funciones sustitutas" suelen expresarse en términos del *margen*, un término que se explica en el Apartado 4.3, por no ser exclusivo de los ensembles tipo Boosting.

Estrategias mixtas de promoción de la diversidad

En el estudio se han usado dos ensembles híbridos entre Bagging y Boosting, Iterated Bagging y Regresión Aditiva, ambos para problemas de regresión. Los ensembles híbridos fueron diseñados para mejorar dos magnitudes (*sesgo* y *varianza*), encuadradas dentro de lo que se conoce como *componentes de error de un predictor*, concepto que se detallará en el Apartado 4.2 y se explica aquí someramente. La definición de estas componentes parte de considerar la clase predicha por un sistema de predicción como una variable aleatoria. Esto es, un mismo predictor puede predecir varios valores para unos mismos atributos, al haber sido entrenado con conjuntos de datos diferentes. En este contexto, la *varianza* mide la variabilidad entre las diferentes estimaciones y el *sesgo* la diferencia entre la tendencia o valor esperado de las estimaciones y el verdadero valor que se pretende estimar. En un ensemble tipo Bagging se puede reducir la varianza sin alterar sustancialmente el sesgo, mientras que con Boosting la componente que se reduce principalmente es el sesgo, aumentándose en general ligeramente la varianza. A partir de este resultado surgen los dos métodos híbridos, Iterated Bagging y Regresión Aditiva. Mientras que el primero se concibe como una modificación de Bagging para reducir el sesgo además de la varianza, el segundo se plantea como una forma de modificar Boosting para que conseguir una

reducción de varianza.

La idea de Iterated Bagging es usar, además de un predictor inicial tipo Bagging, un segundo predictor que trate de estimar el sesgo. Entonces, la predicción final se construye como la suma de la predicción inicial más la corrección para compensar el sesgo. En la ecuación 3.93 se muestra esquemáticamente la corrección, siendo y el valor real de la clase y $f_e(x)$ el valor estimado.

$$sesgo = E\left(y - f_e(x)\right) \rightarrow y = f_e(x) + \text{Estimacion}(sesgo) \tag{3.93}$$

La forma de estimar el sesgo es iterativa a través de los *residuos*, las diferencias entre el valor real y el estimado ($y - f_e(x)$, en la primera iteración). Entonces, se van modificando los conjuntos de entrenamiento que se usan en cada iteración. Los datos de entrenamiento del primer predictor, T, usan los valores iniciales de la clase, y, mientras que para el segundo se toman los mismos atributos pero cambiando la clase por los residuos ($y - f_e(x)$), resultando un nuevo conjunto de entrenamiento, T'. Al igual que se ha definido T' a través de los residuos de un modelo entrenado con los datos T, se puede continuar de forma iterativa, tomando un segundo conjunto T'' a partir de los residuos de T', un tercer conjunto T''' a partir de los residuos de T'', etc, resultando la formulación del predictor expresada en 3.94. El resultado es que la modificación de Bagging acaba derivando en un aprendizaje secuencial como en Boosting, ya que cada predictor base aprende donde los anteriores han tenido mayor error (mayor *residuo*, en la notación del método). Dado que el residuo es el gradiente del error cuadrático, el entrenamiento es asimilable al de un ensemble AdaBoost en la formulación de Gradient Boosting y con función de pérdida cuadrática. Podría parecer entonces que Iterated Bagging es un ensemble AdaBoost que tiene ensembles tipo Bagging como predictores base. Sin embargo, no es exactamente así, ya que el cálculo de residuos tiene en cuenta el muestreo de Bagging. Así, para cada instancia, no se consideran en dicho cálculo los predictores base de Bagging que fueron entrenados con la instancia.

$$y = f_e(x,T) + f_e(x,T') + f_e(x,T'') + f_e(x,T''') + \ldots \tag{3.94}$$

El ensemble que hemos llamado Regresión Aditiva es la implementación de Weka [203] del método de aprendizaje Stochastic Gradient Boosting. Dicho método de aprendizaje es una modificación de Gradient Boosting destinada a disminuir el posible sobreajuste

originado por éste. Para ello, toma la estrategia de Bagging de entrenar cada predictor base en un subconjunto de los datos. El resultado final es similar al del ensemble Iterated Bagging, como se explica en [181], ya que en ambos se usan los residuos para entrenar modelos sucesivos y se aplica el remuestreo de Bagging. Sin embargo, en Iterated Bagging se entrena un ensemble Bagging cada vez y en la Regresión Aditiva no hay como tal ningún ensemble Bagging, sino un ensemble Boosting modificado con el muestreo de Bagging.

3.6. Clasificación Ordinal

La clasificación ordinal es un problema de predicción en el que la clase puede tomar un conjunto finito de valores entre los que existe una relación de orden. Por ejemplo, clasificar la calidad de una pieza como baja, media o alta. Dentro de los tipos de problemas que aborda la Minería de Datos, explicados en 3.1, encaja formalmente con la definición de clasificación, puesto que la clase que se desea predecir toma valores discretos. Sin embargo, no es conveniente analizar este tipo de problemas directamente por una técnica de clasificación, ya que éstos asumen que no hay relación de orden en las clases [66]. Esto haría que el clasificador no pudiese distinguir entre grados de error de predicción. Así, no debe ser igualmente penalizado equivocarse al predecir en vez la clase real su valor inmediatamente posterior o anterior que un error de predicción entre clases muy alejadas. Retomando el ejemplo de la calidad de una pieza, no es igualmente grave confundir una pieza de baja calidad con una de calidad media que con una de calidad alta.

El problema de clasificación ordinal está a medio camino entre la clasificación y regresión, por lo que se ha resuelto desde ambos enfoques. Así, una de las soluciones más habituales es definir un problema de regresión y luego discretizar la clase predicha [111]. Por otro lado, también es posible usar técnicas de clasificación forzando a que se tenga en cuenta el orden entre clases. Un método para aportar dicha información a los clasificadores es transformar el problema inicial en varios problemas binarios, en los que se utilice cada uno para predecir si la clase ha alcanzado un determinado valor. De esta forma, un problema k clases se transforma en $k-1$ problemas binarios [66], donde el i-ésimo predice si la clase es mayor que el valor ordinal i-ésimo, tal como se muestra en la Figura 3.13. Por tanto, hay que entrenar *k-1* clasificadores binarios, combinando la probabilidad que

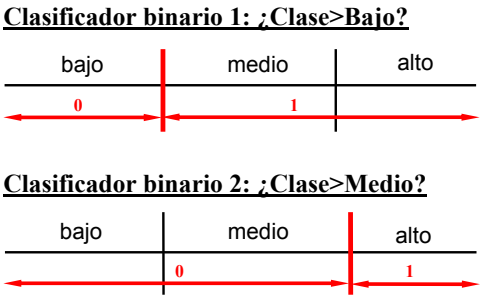


Figura 3.13: Formación de clases binarias en la clasificación ordinal

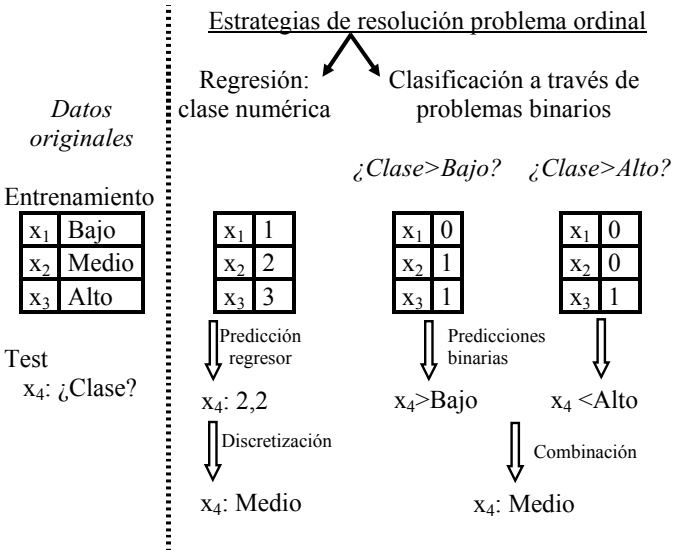


Figura 3.14: Estrategias de resolución de la clasificación ordinal

estiman para definir la predicción final. En la Figura 3.14 se resumen las dos estrategias expuestas para resolver la clasificación ordinal, utilizar un regresor y discretizar o bien tomar clasificadores binarios.

El artículo "*Improvements in modelling of complex manufacturing processes using classification techniques*" (sección 5.1) recoge las técnicas de clasificación ordinal consideradas en el presente estudio.

3.7. Clasificación Desequilibrada

Se dice que se tiene un problema de clasificación desequilibrado cuando hay una clase para la que se tienen la mayoría de las instancias, frente al resto, llamadas *minoritarias*, con menor porcentaje de instancias [73]. En este contexto deja de ser válida la métrica habitual para medir el rendimiento de los clasificadores, el porcentaje de instancias correctamente clasificadas. Para ilustrar este hecho, puede considerarse un conjunto de datos con 1000 instancias de dos clases: la clase A con 990 instancias, y la B, con 10 instancias. Un clasificador trivial que siempre predijera la clase A para cualquier instancia, sin importar cuál fuera en realidad su clase, tendría un porcentaje de acierto esperado del 99 %. Sin embargo, no suministraría ninguna información útil, ya que la *clase minoritaria* no sería tenida en cuenta. En otras palabras, la clase B podría ser indetectable con clasificadores diseñados para resolver problemas de clasificación general y no específicamente de problemas desequilibrados.

Por tanto, en este tipo de problemas de clasificación es conveniente usar tanto técnicas de clasificación como métricas específicas. A continuación se explican ambos aspectos, pudiéndose consultar las técnicas de clasificación más en detalla en el artículo "*Identifying maximum imbalance in datasets for fault diagnosis of gearboxes*" (sección 5.2). En dicho artículo, sin embargo, por limitaciones de espacio no se aclararon en dicha publicación algunas particularidades de ciertos algoritmos que se usan en algunas de las técnicas de clasificación multiclase desequilibrada. Específicamente, no se profundizó sobre técnicas usadas frecuentemente para resolver el problema de clasificación desequilibrada binario. Hay que tener en cuenta que sí que existe un consenso sobre cuáles son las estrategias más adecuadas para clasificación desequilibrada cuando el problema sólo tiene dos clases [72], a diferencia del caso multiclase en el que se han publicado trabajos que siguen enfoques diferenciados [74, 83, 87, 211, 216]. Así, para el caso binario es frecuente usar unos métodos de remuestreo de los datos para transformar el problema en uno equilibrado. Dichos métodos se detallan en 3.7.2, explicando posteriormente los métodos de clasificación para problemas desequilibrados binarios, tanto los que hacen uso de las técnicas de remuestreo como otros clasificadores.

Cuadro 3.7: Matriz de confusión en problemas binarios

		Valor predicho	
		Predicción positiva	**Predicción negativa**
Valor real	**Instancias positivas**	Verdadero positivo (VP)	Falso negativo (FN)
	Instancias negativas	Falso positivo (FP)	Verdadero negativo (VN)

3.7.1. Métricas para problemas de clasificación desequilibrada

Dado que las métricas para problemas de clasificación desequilibrados multiclase se definen a partir de las métricas para clasificación desequilibrada binaria, comenzamos por explicar las métricas de problemas binarios. Los problemas de clasificación binaria tienen dos clases: una con la mayor parte de la instancias, i.e., la *clase mayoritaria* y otra con un pequeño porcentaje de las instancias, i.e., la *clase minoritaria*. En este contexto, se cambia la notación de las clases a *clase positiva* (o *información relevante*) para la *clase minoritaria*, y *clase negativa* para la *clase mayoritaria*, enfatizando que también es necesario una estimación precisa de la *clase positiva*, y no solamente de la clase negativa.

A partir de los conceptos de clase *positiva* y *negativa*, es posible distinguir entre dos tipos de errores que puede cometer un clasificador, definiendo lo que se conoce como *matriz de confusión* (Cuadro 3.7), y, consecuentemente, los valores de *precisión (P)* y *exhaustividad (E)* (conocido como *recall* en la bibliografía inglesa), que se indican en las ecuaciones 3.95:

$$\begin{cases} \text{precision} = P = VP/(VP+FP) = VP/(predicciones\,positiva) \\ \text{exhaustividad} = E = VP/(VP+FN) = VP/(instancias\,positivas) \end{cases} \tag{3.95}$$

Comparando las expresiones de precisión y exhaustividad, ambos valores pueden ser vistas como un ratio de verdaderos positivos, pero mientras que la precisión se centra en los valores predichos, la exhaustividad se centra en los valores reales. En otras palabras, con la precisión el objetivo se expresa en términos de fiabilidad de la predicción (¿cuántas de las instancias predichas como positivas realmente lo son?), mientras que en el caso de la exhaustividad, el objetivo se expresa en términos de sensibilidad (¿cuántas instan-

cias positivas puede detectar el clasificador?). En ambos casos, los verdaderos positivos son importantes para obtener valores altos de precisión y exhaustividad. Sin embargo, una buena precisión implica pocos falsos positivos, mientras que es el número de falsos negativos lo que debe ser bajo en el caso de la exhaustividad. A la hora de ponderar cuál de los dos valores es más importante, la pregunta puede ser reformulada como: "¿qué error es más preocupante un falso positivo o un falso negativo?".

Desde un punto de vista industrial, las clases positivas representan fallos en máquinas y las negativas condiciones de funcionamiento normal. Entonces, si la situación que implica más coste es parar la máquina cuando no se iba a producir ningún fallo (i.e., un falso positivo), el clasificador debería centrarse en la precisión. Por otro lado, si el peor caso implica no parar la máquina cuando se va a producir un fallo (i.e., un falso negativo), la meta debe ser maximizar la exhaustividad.

En general son importantes tanto la precisión como la exhaustividad, por lo que se define un conjunto de métricas que combinan ambos valores. Se explican a continuación dos métricas populares para la clasificación binaria: el valor F (*F-measure*, en la bibliografía inglesa), y el índice de correlación de Matthews, MCC. Además de estas medidas, el área bajo la curva ROC, AUC, es también popular para problemas binarios, pero no existe una generalización clara para problemas multiclases, por lo que no se describe aquí.

El valor F es la media armónica de precisión y exhaustividad. Como la media aritmética, la media armónica proporciona un sistema de promediado de un conjunto de números, aunque utiliza los valores inversos. Se define esta media como la inversa de la media de las inversas. Para el caso del valor F, la media armónica de precisión y exhaustividad viene dado por la ecuación 3.96. La principal diferencia entre la media armónica y la convencional es que los valores pequeños hacen que baje más drásticamente la media en el primer caso. De hecho, el valor F sería cero cuando la precisión fuera 1 y la exhaustividad 0, mientras que la media aritmética sería 0,5.

$$F = \frac{1}{\frac{1}{2}(\frac{1}{P} + \frac{1}{E})} = \frac{2PE}{P+E} \tag{3.96}$$

Otra forma de estimar el rendimiento de un clasificador es mediante MCC, una métrica basada en conceptos estadísticos. Para definir esta medida, se asimilan los datos y la predicción con variables estadísticas, D y M, respectivamente. Entonces, puede usarse también la idea de correlación, frecuentemente usada en estadística para definir el grado

Cuadro 3.8: Matriz de confusión en términos de D y M

	M	$\bar{M}$
D	VP	FN
$\bar{D}$	FP	VN

de relación entre dos variables (Ecuación 3.97).

$$\begin{cases} MCC = \frac{cov(D,M)}{\sqrt{cov(D,D)cov(M,M)}} \\ cov(X,Y) = media[(X - media(X))\,(Y - media(Y))] \end{cases} \tag{3.97}$$

Una vez que se han definido las dos variables, D y M pueden formularse como vectores binarios 0/1 para el caso de clasificación binaria (tomando 1 para la clase positiva). En este caso particular, MCC puede interpretarse más fácilmente a partir de la matriz de confusión, cuya estructura se ilustra en la Tabla 3.8.

Se puede calcular la covarianza a partir de los valores de la matriz de confusión [8], como se indica en la ecuación 3.98.

$$\begin{cases} cov(D,M) = TP \cdot TN - FP \cdot FN \\ cov(D,D) = (TP + FN) \cdot (FP + TN) \\ cov(M,M) = (TP + FP) \cdot (FN + TN) \end{cases} \tag{3.98}$$

La expresión final de MCC para problemas binarios viene dada por la Ecuación 3.99.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FN) \cdot (FP + TN) \cdot (TP + FP) \cdot (FN + TN)}} \tag{3.99}$$

Una vez vistas las métricas para el caso binario, pasamos al caso multiclase. En este trabajo se analizan tres métricas para problemas de clasificación multiclase desequilibrada: la macro-media F, $MacroF$, la media $G-mean$ y MCC. La métrica seleccionada como principal, $MacroF$, generaliza el concepto de valor F a múltiples dimensiones, promediando los valores F para cada par de clases, como se muestra en la Ecuación 3.100 para un problema con m clases.

$$MacroF = \frac{\sum_{i=1}^{m} F_i}{m} \tag{3.100}$$

La segunda métrica analizada es $G-mean$. Esta métrica utiliza la media geométrica

como sistema de promediado de la precisión obtenida para cada clase, definida como el ratio de instancias correctamente clasificadas para cada clase. Este concepto es similar al de *exhaustividad* usado para el caso binario.

$$G-mean = \left[\prod_{i=1}^{m} \frac{\text{Predicción correcta } i}{\text{Número total de instancias } i}\right]^{1/m} \tag{3.101}$$

Finalmente, se detalla *MCC*. La definición de este índice para el caso multiclase es un tema de investigación abierto [100]. La formulación usada en el presente trabajo consiste en una media ponderada de los valores de *MCC* obtenidos para cada clase (MCC_i). Para un problema con m clases, N_i instancias de la clase i-ésima y N instancias totales, *MCC* se calcula la ecuación 3.102. En esta expresión, si una clase tiene casi todas las instancias (i.e., $N_i \to N$), el valor de *MCC* tiende a ser igual que el valor de la métrica calculado para la clase mayoritaria (i.e., $MCC \to MCC_i$). A modo de ejemplo, se ha usado el Cuadro 3.9 para ilustrar las diferencias entre las métricas *MCC* y $MacroF$.

$$MCC = \frac{\sum_{i=1}^{m} N_i \times MCC_i}{N} \tag{3.102}$$

En el Cuadro 3.9 se seleccionó un ejemplo de test con tres clases y 1000 instancias "OK" (funcionamiento correcto de la máquina), 50 instancias del fallo tipo "I" y 10 instancias del fallo tipo "II". Se proponen tres matrices de confusión (A, B y C) para comparar el rendimiento de las dos métricas. Se toma el Caso A como referencia, mientras que B y C se usan para analizar cómo se ven afectadas las métricas por cambios en los términos de error. El número total de instancias incorrectamente clasificadas es el mismo en los casos A y B (10 instancias), pero mientras que en A la distribución del error entre las clases "OK" y "I" es simétrico, en B todos los fallos están en un único término de la matriz de confusión. Tanto *MCC* como $MacroF$ en la matriz de confusión B descienden un 0.4 % en comparación con la matriz de confusión A, de modo que en el ejemplo ambas métricas tienen la misma sensibilidad a la asimetría en la distribución de error. El Caso C se escoge para ilustrar cómo se ven afectadas las métricas por el bajo rendimiento respecto de una clase minoritaria concreta. Mientras que *MCC* no se ve afectada por los cambios en la matriz de confusión, la variación de $MacroF$ es considerable (13.8 %). Puede concluirse que $MacroF$ es más adecuada para evaluar el rendimiento de un clasificador en cada uno de los tipos de fallos considerados en el estudio, ya que evaluar incorrectamente un tipo de

Cuadro 3.9: Sensibilidad al grado de desequilibrio

Caso A
(referencia)

Clasificamo como

	OK	*I*	*II*
OK	995	5	0
I	5	45	0
II	0	0	10

MCC =0.912
$MacroF$ =0.965

Caso B
(error asimétrico en I)

Clasificado como

	OK	*I*	*II*
OK	1000	0	0
I	10	40	0
II	0	0	10

MCC =0.908 ($\downarrow$0.4 %)
$MacroF$ =0.961 ($\downarrow$0.4 %)

Caso C
(bajo rendimiento en II)

Clasificado como

	OK	*I*	*II*
OK	995	0	5
I	0	50	0
II	5	0	5

MCC =0.912 (=)
$MacroF$ =0.832 ($\downarrow$13.8 %)

fallo de un aerogenerador sería inaceptable desde un punto de vista industrial. Los fallos excepcionales pueden ser mucho más críticos que los más comunes y deben ser identificados adecuadamente. De todos modos, el rendimiento de los mejores clasificadores para conjuntos desequilibrados también se evalúa con la métrica *MCC* en esta tesis, ya que es una métrica muy popular que facilita la comparación con las investigaciones de otros autores. La información que proporciona *MCC* es también más conveniente que $G-mean$ en nuestro caso, ya que podrían pederse objetivos de optimización en términos de *precision* usando $G-mean$.

3.7.2. Técnicas de remuestreo de los datos

Dado un conjunto inicial con una serie de clases, una mayoritaria y otras minoritarias, hay dos formas de conseguir la misma frecuencia para todas las clases: o bien se reduce el número de instancias de la clase mayoritaria, lo que se conoce como *undersampling*, o bien se generan artificialmente más instancias de las clases minoritarias, técnica que recibe el nombre de *oversampling*. En el presente trabajo se ha usado el método *Synthetic Minority Over-sampling Technique (SMOTE)* [46] para hacer *oversampling*, mientras que para realizar *undersampling* simplemente se seleccionan las muestras necesarias de la clase mayoritaria de forma aleatoria, lo que se denomina *Random Under Sampling* [11].

En SMOTE se generan instancias sintéticas de las clases minoritarias. Se distingue entre estas instancias sintéticas y las que originalmente presenta el conjunto de datos que

llamaremos *instancias iniciales*. Las instancias sintéticas generadas se sitúan en el espacio de atributos en un punto intermedio entre dos instancias de las iniciales pertenecientes a la misma clase. No se considera necesariamente para cada instancia inicial que su vecina es el punto más cercano, sino que se escoge aleatoriamente una instancia de entre las k más cercanas ($k = 5$ en [46]). A la hora de determinar qué dos instancias iniciales forman el segmento para definir la instancia sintética, se toman una serie de instancias iniciales como referencia. De este modo, a partir de una misma instancia inicial de referencia se pueden calcular varias instancias sintéticas, seleccionado vecinos de la instancia de referencia tantas veces como instancias sintéticas haya que calcular. Inicialmente, se calcula cuántas instancias de referencia hay que tomar y cuántas instancias sintéticas se generarán por cada instancia de referencia. Por ejemplo, si es necesario generar un 200 % más de instancias, todas las instancias de la clase minoritaria se usarían de referencia, obteniendo dos a partir de cada una de ellas. Para ello, se seleccionaría dos veces uno de los k puntos más cercanos a cada referencia. Sin embargo, si sólo se quisiera generar un 20 % más de instancias, no se tomarían todas las instancias de la clase minoritaria como referencia sino sólo el 20 %, generando una sólo instancia sintética por referencia.

Dado un par de instancias iniciales, la forma de generar una instancia sintética en un punto intermedio en el espacio de atributos es usar un número aleatorio entre 0 y 1. De esta forma, si llamamos A y B a los puntos iniciales y r al número aleatorio, el punto generado, C, viene dado por 3.103.

$$C = A + r(B - A) \tag{3.103}$$

En la Figuras 3.15 se muestra el funcionamiento de SMOTE, indicando el conjunto de datos inicial a la izquierda, de dos dimensiones, y el resultado de aplicar la técnica de remuestreo a la derecha. Se usan círculos azules para graficar la clase mayoritaria, cuadrados rojos para las instancias iniciales de la clase minoritaria, y cuadrados naranjas para las instancias sintéticas generadas.

3.7.3. Técnicas de clasificación desequilibrada binaria

En el presente trabajo se han usado las cuatro técnicas de clasificación desequilibrada binaria más populares [72]: SMOTEBagging [198], SMOTEBoosting [47], RUSBoost

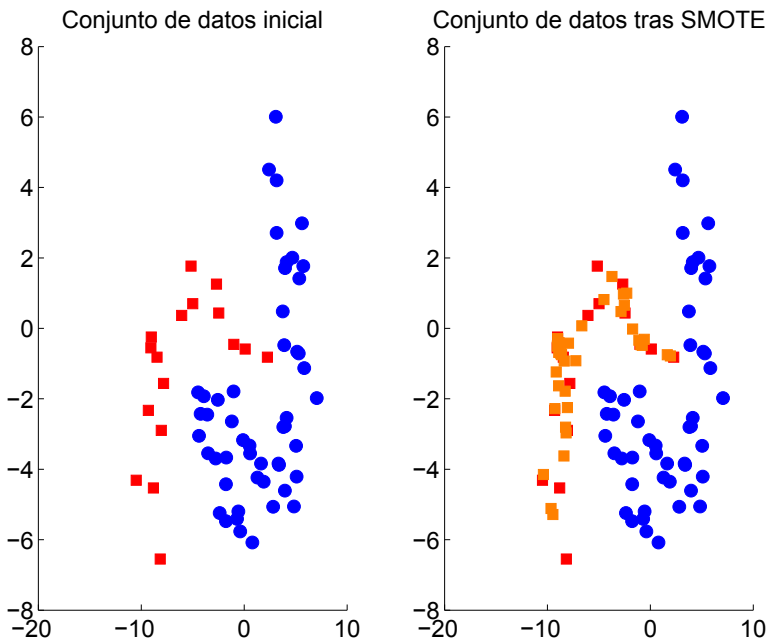


Figura 3.15: Funcionamiento de SMOTE

[171] y C4.4 [145]. Las tres primeras técnicas se explican a partir de conceptos vistos anteriormente, ya que son ensembles (Apartado 3.5.5) construidos sobre modificaciones del conjunto de datos desequilibrado hasta transformarlo en uno equilibrado (Apartado 3.7.2). SMOTEBagging y SMOTEBoosting usan en el remuestreo SMOTE, siguiendo una estructura tipo Bagging y Boosting, respectivamente, mientras que en RUSBoost el esemble es tipo Boosting y el remuestreo es Random Under Sampling.

El método C4.4 requiere sin embargo una explicación más detallada. El desarrollo de este método fue motivado por limitaciones observadas en el método C4.5, tal como se indicó en 3.5.1. Antes de explicar dichas carencias, es necesario aclarar que las técnicas de clasificación no sólo predicen la clase de la instancia a clasificar, sino que la mayoría de algoritmos pueden además indicar con qué grado de certeza hacen esa consideración, a través de una asignación de probabilidades a cada una de las clases. La limitación de C4.5 reside en la estimación de las probabilidades de cada clase, ya que se ha determinado que dicha estimación es poco precisa, aún para conjuntos en los que alcanzan gran porcentaje de acierto. Para ilustrar esta situación recordemos que el método se basa en hacer divisiones sucesivas hasta tener un árbol con las hojas lo más homogéneas posibles (de una sola clase). A cada clase se le asigna una probabilidad igual al ratio de instancias de

dicha clase en la hoja. Pero entonces, si el número de instancias que quedan en cada hoja es muy pequeña, C4.5 tenderá a asignar a las instancias que clasifica a la clase mayoritaria de la hoja con el 100 % de probabilidad. Sin embargo, la certeza real de pertenencia a la clase asignada es pequeña, puesto que hay pocos ejemplos en la hoja para confirmar esa situación. Por ello, se propusieron dos alternativas, o bien acompañar la información de la clase estimada con información referente a cuántas instancias formaron la hoja que proporciona dicha clasificación o bien *suavizar* la estimación de probabilidad, caso en el que se engloba el clasificador C4.4.

El problema de suavizar la probabilidad cuando se tiene un número de muestras pequeño no sólo se ha aplicado a la Minería de Datos, sino que es un problema ampliamente estudiado por la Estadística [177]. La solución del problema propuesta en C4.4 es la corrección de Laplace, que introduce la probabilidad a priori de tener cada clase. Esto quiere decir que si no tuviéramos ningún conocimiento de los datos y sólo supiéramos el número de clases del problema a considerar, C, todas las clases deberían ser equiprobables. Así, según la corrección de Laplace cuando hay cero instancias la probabilidad es $1/C$, y la probabilidad sólo podría alcanzar exactamente 1 con un conjunto que tuviera un número infinito de instancias, N, y que además ese número infinito de instancias estuviera en la hoja del árbol cuya probabilidad de predicción se está estimando. Si k es el número de instancias en dicha hoja, entonces $N = k \to \infty$. La expresión 3.104 muestra la probabilidad, p, a estimar según la corrección de Laplace.

$$p = \frac{k+1}{N+C} \tag{3.104}$$

En [145], Provost y Domingos estudiaron la estimación de probabilidades de C4.4 en sí, y no su aplicación a problemas de clasificación binaria desequilibrada. Sin embargo, sus conclusiones son especialmente aplicables en el problema estudiado en el presente trabajo por dos razones. Por un lado, los árboles se usan como clasificadores base dentro de ensembles, los cuáles construyen la clasificación a partir de un voto ponderado de las clasificaciones de cada clasificador base. En algunos tipos de ensemble estas ponderaciones están relacionadas con la estimación de probabilidad, por lo que determinan de forma crucial la precisión del ensemble. Por otro lado, la distribución de las clases en los problemas desequilibrados los hace más susceptibles de originar árboles con hojas con pocas instancias, causa que dio lugar a la corrección planteada por el método. Hay que tener en cuenta

que se tienen pocas instancias de la clase minoritaria en comparación con la mayoritaria, por lo que es más probable que éstas queden en hojas aisladas.

Como hemos visto en 3.5.1, es habitual aplicar dos sistemas de poda a los árboles de decisión con el fin de evitar que sobreajusten a los datos de entrenamiento. Sin embargo, los experimentos de Provost y Domingos sobre los árboles C4.4 con la corrección de Laplace para la estimación de la probabilidad muestran una mayor precisión sin la poda. Por ello, lo que se conoce como C4.4 es una modificación de C4.5 con los tres cambios anteriormente citados: no hay poda ni pre-poda ni post-poda, y la estimación de probabilidad se hace con la corrección de Laplace.

Capítulo 4

Fundamentos teóricos de los ensembles

Existen varias teorías para explicar de los buenos resultados experimentales obtenidos por predictores de tipo ensemble. La más extendida es la basada en la descomposición de error en sesgo y varianza, pero también existen otras teorías para explicar algunos ensembles concretos, como la teoría del margen aplicada a Adaboost, especialmente cuando éste se utiliza para clasificación [93]. A continuación se explican ambas justificaciones teóricas. Para el caso de la descomposición del error, es válido para todos los ensembles y tanto para clasificación como para regresión, aunque existe un mayor consenso para regresión, habiendo diversos enfoques en clasificación. Por otro lado, para el caso de la teoría del Margen el análisis sólo se refiere a Adaboost.

4.1. Descomposición del error Sesgo-Varianza

Como se indicó en 3.4.3, para escoger un sistema de predicción preciso hay que buscar un equilibrio entre modelos excesivamente sencillos y modelos excesivamente complejos. Así, en el ejemplo de la regresión polinómica de la Figura 3.6, se observaba como órdenes de los polinomios bajos llevaban a una *falta de ajuste*, mientras que para órdenes altos se tenía un *sobreajuste*. En esta sección se explican dos términos, *sesgo* y *varianza*, que pueden cuantificar en qué situaciones hay *falta de ajuste* o *sobreajuste*. Así, decimos que un modelo tiene falta de ajuste cuando el sesgo es alto y que hay sobreajuste cuando la varianza es elevada.

Los conceptos de sesgo y varianza no son exclusivos de Minería de Datos, sino que se

definen dentro de la Estadística Inferencial (introducida en 3.3), y más específicamente de la Estimación de Parámetros de una variable aleatoria. En este contexto, en la bibliografía clásica de Teoría Estadística expresan la calidad de un predictor, $\hat{\theta}$, del verdadero parámetro de una variable aleatoria, θ, en términos de su sesgo y su varianza. En [204] se indica que el error cuadrático medio, MSE, es útil para determinar si $\hat{\theta}$ es una buena estimación de θ y que dicho error puede descomponerse en dos términos, la varianza y el cuadrado del sesgo, como indica la ecuación 4.1.

$$\begin{aligned} &MSE\left(\hat{\theta}\right) = Varianza\left(\hat{\theta}\right) + Sesgo^2\left(\hat{\theta}\right) \\ &MSE = E\left(\left[\hat{\theta} - \theta\right]^2\right) \\ &Varianza = E\left[\left(\hat{\theta} - E\left[\hat{\theta}\right]\right)^2\right] \\ &Sesgo = E\left[\hat{\theta}\right] - \theta \end{aligned} \tag{4.1}$$

De acuerdo a la ecuación 4.1, se pueden establecer las siguientes definiciones:

- **Sesgo**. **Diferencia** entre el valor promedio **estimado** y el **verdadero valor** que se pretende estimar.
- **Varianza**. Medida de la **variabilidad de estimaciones** sucesivas que indica cuál es el promedio de la diferencia entre cada estimación y el valor promedio de todas ellas.
- **Error cuadrático medio**. Valor promedio esperado del **cuadrado de la diferencia** entre el valor real del parámetro y el predicho.

El problema de predicción estudiado por la Minería de Datos es más complejo que el de estimación de parámetros de variables aleatorias analizado por la Estadística Inferencial, y para el cual se definieron los conceptos de sesgo y varianza. Hay que tener en cuenta que un parámetro es un único valor, mientras que el objetivo de un sistema de predicción es pronosticar múltiples valores. Por ejemplo, un problema de estimación de parámetros podría ser determinar la vida media de un grupo de individuos, mientras que un regresor trataría de predecir la esperanza de vida de cada uno de ellos a través de una serie de atributos. En este contexto, las definiciones de sesgo y varianza mostradas anteriormente son útiles para entender qué quieren expresar dichos conceptos y cómo pueden relacionarse con la falta de ajuste o el sobreajuste, pero deben generalizarse para los problemas de clasificación y regresión.

A continuación se explica cómo se extienden los conceptos de sesgo y varianza a clasificación y regresión. En primer lugar se analiza la descomposición para el error cuadrático medio y posteriormente para una pérdida cualquiera y para definiciones específicas de regresión. Pero, antes de pasar a esos desarrollos, se da una explicación más intuitiva de dichos componentes del error que permite fijar más claramente qué es lo que se quiere cuantificar con cada magnitud, lo que puede resultar especialmente útil para el caso de problemas de clasificación, en los que existen varias definiciones en la literatura de Minería de Datos. Otra diferencia entre la definición vista para Estadística Inferencial y la definición para Minería de Datos es que ésta última considera el ruido. Así, dados unos atributos fijos el valor real de la clase no siempre es el mismo. Siguiendo con el ejemplo, sean cuales sean las variables que consideremos para predecir la esperanza de vida de una persona, habrá dos personas que aún con los mismos atributos tengan una esperanza de vida distinta. No importa si esto se produce por no considerar los atributos adecuados o no, el hecho es que en general los problemas tratados en Minería de Datos no son resolubles a la perfección (no se puede predecir con total certeza la clase en función de los atributos, independientemente de cuán buena sea la técnica de predicción).

4.1.1. Visión intuitiva del sesgo y la varianza

Una forma visual de entender los conceptos de sesgo y varianza es asimilar el problema de estimar un parámetro con jugar a los dardos, donde el verdadero valor del parámetro es el centro de la diana y cada estimación se corresponde con una tirada [164]. Se han tomado dos figuras para comparar ambas fuentes de error con esta analogía. En ellas, el objetivo de la predicción es el centro de una diana, y los valores predichos por el modelo se representan por los hexágonos negros. En la Figura 4.1, la componente que aparta al error es la varianza, mientras que en la Figura 4.2, el error es debido fundamentalmente al sesgo.

Cuando un predictor tiene error debido a la varianza pero no al sesgo, las predicciones son muy variables, siendo en promedio similares al verdadero valor a estimar, pero pudiendo estar una predicción individual alejada del mismo. En el caso opuesto, si el error se debe al sesgo y no a la varianza, todas las predicciones están muy próximas entre sí, pero alejadas del verdadero valor.

La analogía de la diana no es completamente válida para los problemas de Minería de

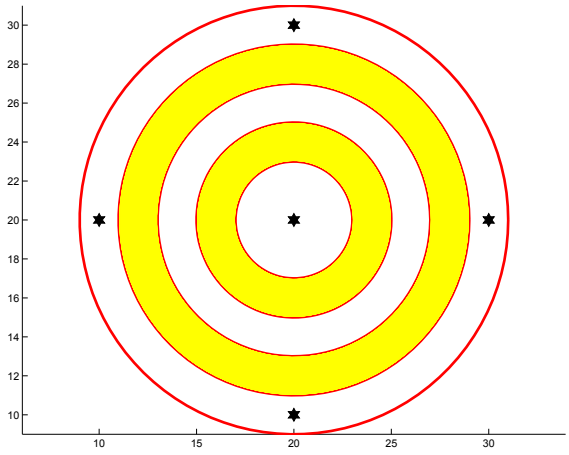


Figura 4.1: Error sin sesgo pero con alta varianza

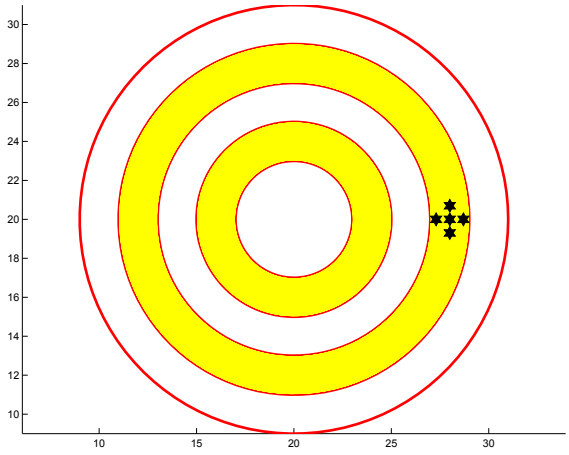


Figura 4.2: Error con alto sesgo y baja varianza

Datos debido al ruido. Como se ha explicado, no es posible que un predictor acierte siempre, no siempre se podría dar al centro de la diana. Es posible asimilar esta modificación con que la diana se moviera para algunas de las tiradas, pudiendo perjudicar o mejorar la tirada.

Como resumen de la idea de los conceptos vistos, se tienen las siguientes relaciones:

- Varianza - Diferencia entre varias predicciones.
- Sesgo - Distancia de la media de las predicciones al valor real.
- Ruido - Variabilidad del valor real.

4.1.2. Formulación introductoria (Sin considerar ruido)

Una vez explicados los conceptos de sesgo y varianza, se desarrolla la descomposición del error en componentes para el caso de regresión. Se van a realizar dos descomposiciones. En primer lugar, sin considerar que en los datos a predecir hay ruido, y posteriormente incluyéndolo. La primera descomposición es la más sencilla y es formalmente idéntica a la que se usa en Estadística Inferencial, que se ha detallado anteriormente. La segunda descomposición, sin embargo, se prefiere para el caso de Minería de Datos [61]. No obstante, ambas descomposiciones son matemáticamente igualmente válidas. El término llamado sesgo en la primera descomposición se divide en dos términos en la segunda descomposición. Por tanto, ambas descomposiciones comparten la idea de varianza de un estimador, pero difieren en la de sesgo. Para distinguir ambas notaciones, se usa $sesgo_*$ en la primera descomposición y *sesgo* (a secas) en la segunda. Es esta segunda definición la que se ha tomado como referencia en el presente estudio.

La notación seguida para los desarrollos es la siguiente:

- y: Variable de salida que se pretende modelar. *Clase*, en la notación de Minería de Datos.
- x: Variables de las que depende y. *Atributos*, en la notación de Minería de Datos.
- f_e: Modelo estimado para y en función de x. $y \approx f_e(x)$.
- f_r: Relación real entre x e y.

  - Sin considerar ruido en las observaciones, $y = f_r(x)$
  - Considerando ruido en las observaciones, $y = f_r(x) + \varepsilon$

- ε: Ruido en las observaciones. Se considera de tipo *blanco* (con media cero y siguiendo una distribución normal). Su desviación típica es $E\left(\varepsilon^2\right) = \sigma_\varepsilon^2$. Se considera que esta variable no está correlacionada con la clase ni con la estimación de la misma.

- *MSE*: Error cuadrático medio (*Mean Squared Error*). Es la esperanza matemática del cuadrado del error en la estimación, es decir, de la diferencia entre el valor predicho y el real, $MSE = E\left[(y - f_e)^2\right]$.

- $\bar{f}_e$: Media de la estimación.

Introduciendo la media de la estimación, $\bar{f}_e$, en la expresión a la que se la aplica el operador esperanza para calcular el operador *MSE*, se obtiene la descomposición mostrada en 4.2.

$$\begin{cases} (y - f_e)^2 = \left(y - \bar{f}_e + \bar{f}_e - f_e\right)^2 = \left(\left[y - \bar{f}_e\right] + \left[\bar{f}_e - f_e\right]\right)^2 \\ a = y - \bar{f}_e, \;\; b = \bar{f}_e - f_e \\ (a+b)^2 = a^2 + b^2 + 2ab \\ (y - f_e)^2 = \left(y - \bar{f}_e\right)^2 + \left(\bar{f}_e - f_e\right)^2 + 2\left(y - \bar{f}_e\right)\left(\bar{f}_e - f_e\right) \end{cases} \tag{4.2}$$

Aplicando el operador esperanza sobre el desarrollo anterior, se tiene la descomposición del *MSE* indicada en 4.3.

$$MSE = E\left[(y - f_e)^2\right] = E\left[\left(y - \bar{f}_e\right)^2\right] + E\left[\left(\bar{f}_e - f_e\right)^2\right] + 2E\left[\left(y - \bar{f}_e\right)\left(\bar{f}_e - f_e\right)\right] \tag{4.3}$$

Se puede comprobar que el tercer término es nulo, desarrollando el producto (4.4).

$$\begin{cases} \left(y - \bar{f}_e\right)\left(\bar{f}_e - f_e\right) = y\bar{f}_e - yf_e - \bar{f}_e{}^2 + \bar{f}_e f_e \\ E\left[\left(y - \bar{f}_e\right)\left(\bar{f}_e - f_e\right)\right] = E\left[y\bar{f}_e\right] - E\left[yf_e\right] - E\left[\bar{f}_e{}^2\right] + E\left[\bar{f}_e f_e\right] \\ E\left[y\bar{f}_e\right] = E\left[yf_e\right], \;\; E\left[\bar{f}_e{}^2\right] = E\left[\bar{f}_e f_e\right] \\ E\left[\left(y - \bar{f}_e\right)\left(\bar{f}_e - f_e\right)\right] = 0 \end{cases} \tag{4.4}$$

La descomposición de *MSE* en sesgo y varianza queda de la forma indicada en 4.5.

$$\begin{aligned} &MSE = E\left[(y-f_e)^2\right] = E\left[sesgo_*^2\right] + varianza \\ &E\left[sesgo_*^2\right] = E\left[\left(y-\bar{f}_e\right)^2\right],\ sesgo_* = \bar{f}_e - y_r \\ &varianza = E\left[\left(\bar{f}_e - f_e\right)^2\right] \end{aligned} \tag{4.5}$$

Con un cambio de notación, es la misma expresión que 4.1, ya que aún no se ha introducido el ruido.

4.1.3. Formulación de la descomposición para el error cuadrático

Introduciendo el ruido en la formulación, hay que descomponer el valor real de la variable a estimar, y, en dos términos, la relación real que tiene con las variables de entrada, f_r, y el ruido, ε. Al introducir esta descomposición en la expresión del sesgo vista para *MSE* sin considerar ruido queda:

$$\begin{cases} E\left[sesgo_*^2\right] = E\left[\left(y-\bar{f}_e\right)^2\right] \\ \left(y-\bar{f}_e\right)^2 = \left(\left[f_r-\bar{f}_e\right]+[\varepsilon]\right)^2 \\ a = f_r-\bar{f}_e,\ \ b=\varepsilon,\ \ (a+b)^2 = a^2+b^2+2ab \\ E\left[\left(y-\bar{f}_e\right)^2\right] = E\left[\left(f_r-\bar{f}_e\right)^2\right] + E\left[\varepsilon^2\right] + 2E\left[\left(f_r-\bar{f}_e\right)\varepsilon\right] \end{cases} \tag{4.6}$$

El tercer término es nulo debido a que el ruido es independiente de la estimación, por lo que la esperanza del producto se puede calcular como el producto de las esperanzas, siendo nula la esperanza del ruido. Además, se expresa la varianza del ruido como $E\left[\varepsilon^2\right] = \sigma_\varepsilon^2$, con lo que la descomposición del sesgo tal como se definió antes de considerar el ruido es:

$$E\left[sesgo_*^2\right] = E\left[\left(f_r-\bar{f}_e\right)^2\right] + \sigma_\varepsilon^2 \tag{4.7}$$

En la expresión anterior, el primer término es la nueva definición que se hace para el sesgo al considerar el ruido en los datos, y al segundo término se le llama *ruido irreducible*. La descomposición de *MSE* queda:

$$
\begin{aligned}
&MSE = error\,irreducible + E\left[sesgo^2\right] + varianza \\
&error\,irreducible = \sigma_\varepsilon^2 \\
&E\left[sesgo^2\right] = E\left[\left(f_r - \bar{f}_e\right)^2\right],\ sesgo = \bar{f}_e - f_r \\
&varianza = E\left[\left(\bar{f}_e - f_e\right)^2\right]
\end{aligned}
\tag{4.8}
$$

Esta descomposición de *MSE* en tres términos incluye el *error irreducible*, haciendo hincapié en el hecho de que parte del error se debe al propio ruido interno a los datos, sobre el que no se puede actuar, por lo que va a existir siempre independientemente de cuán buena sea la técnica de estimación usada. La definición de sesgo es ahora diferente puesto que no se toma la variable de salida como referencia (y), sino el valor que ésta tendría si no hubiese ruido (f_r).

4.1.4. Descomposición del error para una función de pérdida cualquiera

En el apartado anterior se han definido el sesgo y la varianza como componentes del error cuadrático medio. En el trabajo de James [94], se explica que con este tipo de función de pérdida se admiten dos posibles interpretaciones para el sesgo y la varianza:

- Varianza
  1. Medida de la variabilidad de una estimación.
  2. Error provocado por la variabilidad.
- Sesgo
  1. Diferencia sistemática, esto es, eliminando la variabilidad, entre una estimación y el verdadero valor a aproximar.
  2. Error provocado por la diferencia sistemática entre la estimación y el valor real.

James considera que para una función de pérdida cualquiera pueden usarse las primeras de las interpretaciones para definir *varianza* y *sesgo*, empleando la segunda para la

descomposición del error. Así, llaman *efecto varianza* al error provocado por la variabilidad y *efecto sistemático* al error provocado por sesgo. Para la formulación del sesgo y la varianza se parte ahora del *valor sistemático*, S, de una variable aleatoria respecto de una función de pérdida, L, como aquella cantidad fija que minimiza el valor esperado de la pérdida entre todos los valores de la distribución de la variable aleatoria con respecto de dicha cantidad fija (ecuación 4.9).

$$S(x) = \underset{\mu}{\text{argmín}} E\left[L(x,\mu)\right] \tag{4.9}$$

En los apartados anteriores se han usado también dos términos sistemáticos en la formulación:

- $\bar{f}_e$ - Valor esperado de la clase estimada f_e.
- f_r - Valor que tendría la clase real, y, si no hubiera ruido.

Ahora, se definen S_r y S_e, minimizando L para los valores de y y f_e, respectivamente. Por tanto, S_r desempeña un papel análogo a lo que hasta ahora se ha llamado f_r, mientras que S_e generaliza la función de $\bar{f}_e$.

A modo de ejemplo, se calcula la expresión de S_e para dos funciones de pérdida, suponiendo que se tiene una serie discreta de n estimaciones. Para ello, se iguala a cero la derivada del valor esperado de la pérdida respecto del parámetro μ en la expresión 4.9. En 4.10 se obtiene S_e para el error cuadrático y en 4.11 para el error absoluto. La solución en el primer caso es que S_e sea la media y en el segundo que sea la mediana (el número de muestras mayores que la mediana es igual al número de muestras que son menores que la misma).

$$\begin{cases} E\left[L(x,\mu)\right] = \frac{1}{n}\sum_{i=1}^{n}(x-\mu)^2 \\ \frac{dE}{d\mu} = 0 \;\rightarrow\; \frac{-2}{n}\sum_{i=1}^{n}(x-\mu) = 0 \;\rightarrow\; \sum_{i=1}^{n}(x-\mu) = 0 \\ \sum_{i=1}^{n} x = \mu n \;\rightarrow\; \mu = \frac{1}{n}\sum_{i=1}^{n} x = \bar{x} \end{cases} \tag{4.10}$$

$$\begin{cases} E\left[L(x,\mu)\right] = \frac{1}{n}\sum_{i=1}^{n}|x-\mu| = \frac{1}{n}\left[\sum_{x>\mu}(x-\mu) + \sum_{x<\mu}(\mu-x)\right] \\ \frac{dE}{d\mu} = 0 \;\rightarrow\; \frac{1}{n}\left[\sum_{x>\mu}(-1) + \sum_{x<\mu}(+1)\right] \\ \sum_{x>\mu}(1) = \sum_{x<\mu}(1) \;\rightarrow \mu = \text{Med}(x) \end{cases} \tag{4.11}$$

Cuadro 4.1: Generalización se sesgo y varianza para cualquier L

	Pérdida	
	Cuadrática	General
Varianza	$E\left[\left(\bar{f}_e - f_e\right)^2\right]$	$E\left[L(f_e, S_e)\right]$
Sesgo^2	$\left[\bar{f}_e - f_r\right]^2$	$L(S_r, S_e)$

Así, introduciendo los *valores sistemáticos*, los conceptos de sesgo y varianza se generalizan a cualquier pérdida de acuerdo con el Cuadro 4.1. El término $\bar{f}_e = E(f_e)$, construido con el operador esperanza, E, cuyo estimador muestral es la media, ha sido sustituido por S_e.

A partir de S_e y S_r se definen el *efecto varianza*, EV, y el *efecto sistemático*, ES, según las ecuaciones 4.12 y 4.13, respectivamente.

$$EV = E\left[L(y, f_e) - L(y, S_e)\right] \tag{4.12}$$

$$ES = E\left[L(y, S_e) - L(y, S_r)\right] \tag{4.13}$$

A continuación se comparan dos distribuciones del valor real de la clase, y, para las cuales se tendría el mismo sesgo pero diferentes efectos sistemáticos según se use como función de pérdida el error absoluto o el error cuadrático, considerando que el valor estimado de la clase es constante $y_e = 2$. En ambos casos se tienen 8 instancias repartidas entre los valores 0, 1 y 2. El valor $y = 1$ es el mayoritario para las dos distribuciones, pero representa el 50 % de los casos en la Distribución A frente al 75 % de la Distribución B, como muestra la Figura 4.3. Tanto la media como la mediana de ambas distribuciones es 1, por tanto, S_r es la misma para el error absoluto que para el error cuadrático medio ($S_r = 1 = \bar{y} = med(y)$). Al ser constante la estimación, la media y la mediana coinciden con dicho valor constante, $S_e = 2 = \bar{y}_e = med(y_e)$. En 4.14 se muestran el sesgo y los efectos sistemáticos con el error absoluto como función de pérdida para las dos distribuciones de ejemplo, mientras que los mismos cálculos para una pérdida basada en el error cuadrático se detallan en 4.15.

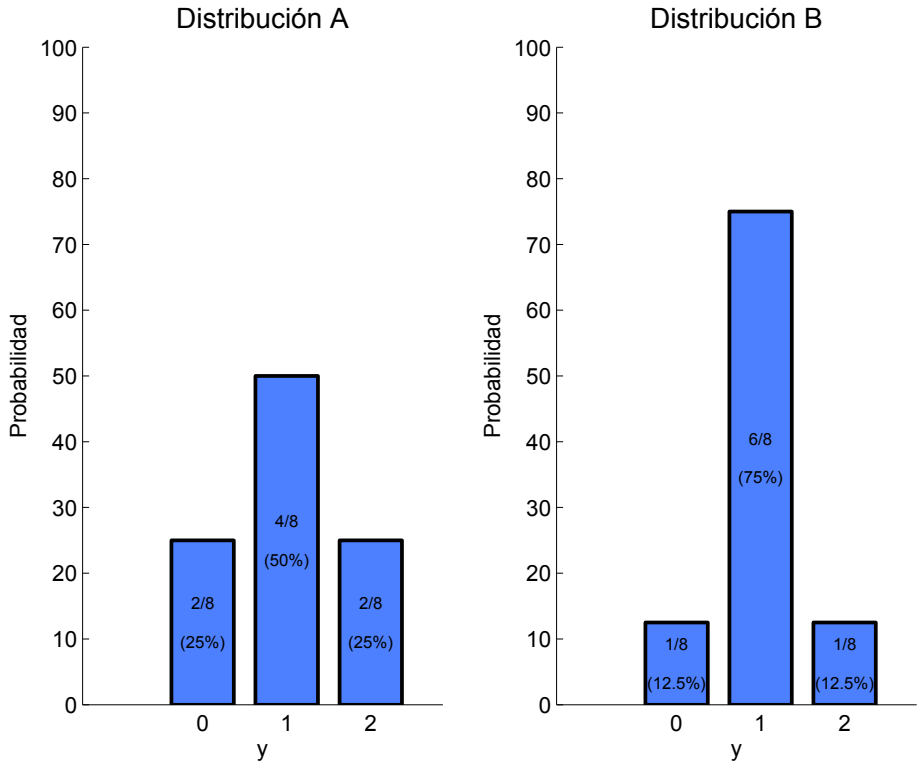


Figura 4.3: Distribuciones de ejemplo para el cálculo del efecto sistemático

$$\begin{cases} ES = E\left[|y - med(y_e)| - |y - med(y)|\right], \ Sesgo^2 = |med(y_e) - med(y)| = 1 \\ ES_A = 2\frac{1}{4} + 1\frac{2}{4} + 0\frac{1}{4} - \left(1\frac{1}{4} + 0\frac{2}{4} + 1\frac{1}{4}\right) = 1 - \frac{1}{2} = \frac{1}{2} \\ ES_B = 2\frac{1}{8} + 1\frac{6}{8} + 0\frac{1}{8} - \left(1\frac{1}{8} + 0\frac{6}{8} + 1\frac{1}{8}\right) = 1 - \frac{2}{8} = \frac{3}{4} \end{cases} \tag{4.14}$$

$$\begin{cases} ES = E\left[(y - \bar{y}_e)^2 - (y - \bar{y})^2\right], \ Sesgo^2 = (\bar{y}_e - \bar{y})^2 = 1 \\ ES_A = 2^2\frac{1}{4} + 1^2\frac{2}{4} + 0^2\frac{1}{4} - \left(1^2\frac{1}{4} + 0^2\frac{2}{4} + 1^2\frac{1}{4}\right) = \frac{6}{4} - \frac{2}{4} = 1 \\ ES_B = 2^2\frac{1}{8} + 1^2\frac{6}{8} + 0^2\frac{1}{8} - \left(1^2\frac{1}{8} + 0^2\frac{6}{8} + 1^2\frac{1}{8}\right) = \frac{10}{8} - \frac{2}{8} = 1 \end{cases} \tag{4.15}$$

Tomando el error absoluto, el efecto sistemático es mayor en la Distribución B, mientras que desde el punto de vista del error cuadrático el efecto sistemático es igual para ambas distribuciones e igual al propio valor del sesgo.

4.1.5. Descomposiciones específicas de problemas de clasificación

Al analizar las fuentes de error de un predictor para un clasificador se prefiere partir de la "**probabilidad** de cometer un error", a diferencia de la regresión, dónde se tomaba directamente "el **error**". En este contexto, se expresa la predicción efectuada por un clasificador a través de un vector de probabilidades asignadas a cada clase, siendo la clase predicha aquella que maximiza la probabilidad. Esta formulación es posible también para aquellos clasificadores que internamente no trabajan con probabilidades, sino que asignan directamente una de las clases. En dichos casos puede considerarse 1 la probabilidad asignada a dicha clase y 0 la asignada al resto de clases.

La analogía usada para regresión de comparar la predicción con jugar a los dardos no es válida ahora. Se tendría un símil más acertado en este caso pensando en que hubiera tantas dianas como clases, pero que al final sólo puntuara una de ellas. Extender los conceptos de sesgo y varianza es entonces complicado, sujeto a diferentes interpretaciones. Por ejemplo, retomando el ejemplo de las dianas, es posible preguntarse qué es mejor, medir el error sólo sobre la diana que puntúa o hacer una medida conjunta sobre todas ellas.

En la literatura de Minería de Datos existen diferentes definiciones de sesgo y varianza para el caso de clasificación, partiendo de una idea común, la probabilidad de cometer un error dada una instancia concreta, x. A partir del vector de probabilidades de cada clase estimadas por el clasificador, al que llamamos p_e, se puede calcular la probabilidad de cometer un error considerando también la probabilidad real de cada clase, a la que denotaremos p_r. El cálculo para instancia, x, viene dado por el complementario de la probabilidad de acierto (la resta a 1 de dicha probabilidad), tal como indica la ecuación 4.16 [112]. La probabilidad de tener un acierto es la suma de las probabilidades de acertar en alguna de las clases, viniendo dada la probabilidad de acierto para una única clase por el producto de las probabilidades reales y estimadas para esa clase.

$$p(error|x) = 1 - \sum_{clases} p_r(x)p_e(x) \tag{4.16}$$

A continuación se comparan las definiciones de sesgo y varianza para una selección de trabajos de entre los que se pueden encontrar en la literatura. Los tres estudios considerados son: Kohavi y Wolpert [108], Breiman [28] y Domingos [61]. Cada descomposición aporta un enfoque diferente. Mientras que la de Kohavi y Wolpert se centra en valores

agregados que tienen en cuenta todas las clases, la de Breiman está más enfocada a la clase esperada. Por otro lado, la descomposición de Domingos permite introducir el concepto de función de pérdida, visto en 3.4.2. Entonces, según este enfoque, existe cierta *subjetividad* a la hora de definir la varianza y sesgo, ya que depende de qué criterio se use para evaluar el rendimiento de un clasificador. Este planteamiento está relacionado con las ideas expuestas en 3.7 para la clasificación desequilibrada. Se explicó entonces que en algunos casos supone más coste predecir erróneamente una clase que otras, por ejemplo un clasificador que detecte fallos de máquinas que se producen con muy poca frecuencia.

En primer lugar, es necesario aclarar algunos aspectos de la notación que se va a seguir, a través del Cuadro 4.2. Una vez vista la notación, la formulación según los tres estudios (denotados por "K" el de Kohavi y Wolpert, "B" el de Breiman y "D" el de Domingos) queda recogida en el Cuadro 4.3, en la que únicamente las definiciones de Domingos hacen uso del concepto de función de pérdida (l). Además, en el Cuadro 4.4 se ha incluido una pequeña explicación de lo que representa cada fórmula. Para el caso de Breiman y Domingos se puede asociar cada término a la probabilidad de un evento aleatorio concreto, pero no para la descomposición de Kohavi y Wolpert, por utilizar agregados de todas las clases. Al igual que para la regresión, se tiene una descomposición en tres términos, pero mientras que en ese caso se usaba el cuadrado del error (a través del MSE), aquí la suma es el propio error (en términos de probabilidad). En las descomposiciones de Kohavi y Wolpert y de Breiman el error es la suma de las tres componentes (sesgo, varianza y ruido), como se indica en la ecuación , pero en la de Domingos no, ya que hay que incluir además unos coeficientes que multiplican al sesgo y la varianza, y que dependen de l. En 4.17 se puede ver dicha diferencia.

$$\begin{aligned} error &= sesgo + varianza + ruido \text{ (Kohavi, Breiman)} \\ error &= c_1(l)sesgo + c_2(l)varianza + ruido \text{ (Domingos)} \end{aligned} \tag{4.17}$$

Se ha añadido para al caso general según la definición de Domingos ("D") el caso particular en el que la función de pérdida es 0/1 ("D2"), para facilitar la comparativa entre divisiones. El rango de variación de los términos en los que hay sumatorios es todas las clases del problema, salvo para el caso en el que se indica dicho rango, la varianza según Breiman (a la que de hecho en su estudio no llama varianza, sino *difusión -spread-*), en la que no se incluye la clase con máxima probabilidad estimada por el clasificador.

Cuadro 4.2: Notación usada en las descomposiciones bias-varianza para clasificación

	Probabilidad	**Rango de valores de la clase, fijado x**	**Valor de la clase que maximiza la probabilidad**
Valores reales	$p_r(x)$	$C_r(x)$	$M_r(x)$
Valores estimados	$p_e(x)$	$C_e(x)$	$M_e(x)$

Cuadro 4.3: Comparación de varias descomposiciones bias-varianza para clasificación

	Sesgo	**Varianza**	**Ruido**
K	$\frac{1}{2}\sum\left[p_r(x)-p_e(x)\right]^2$	$\frac{1}{2}\left[1-\sum p_e^2(x)\right]$	$\frac{1}{2}\left[1-\sum p_r^2(x)\right]$
B	$\left[p_r(M_r\|x)-p_r(M_e\|x)\right]p_e(M_e\|x)$	$\sum_{\neq M_e}\left[p_r(M_r\|x)-p_r(x)\right]p_e(x)$	$1-p_r(M_r\|x)$
D	$l\left(M_r(x),M_e(x)\right)$	$E\left[l\left(M_e(x),C_e(x)\right)\right]$	$E\left[l\left(M_r(x),C_r(x)\right)\right]$
D2	0 (correcto) 1 (incorrecto)	$1-p_e(M_e\|x)$	$1-p_r(M_r\|x)$

Cuadro 4.4: Significado de las descomposiciones

	Sesgo	**Varianza**	**Ruido**
K	*semisuma cuadrado de* **diferencia** entre p. **real** y **estimada** de cada clase	*mitad de 1-sumatorio* **varianza** p. **estimada** cada clase **como** var. **unidimensional**	*mitad de 1-sumatorio* **varianza** p. **real** cada clase **como** var. **unidimensional**
B	**error** en la **estimación** de probabilidad al tomar la **predicción óptima**	**acumulado** del **error** en la **probabilidad estimada respecto** de la clase **real** esperada cuando se predice una clase no óptima	**probabilidad** de tener una **clase distinta** a la **esperada** dados los valores de los atributos
D	**pérdida** valor **real** y predicción **óptima**	**pérdida** esperada de la predicción **óptima respecto** del **resto** de posibles **predicciones**	**pérdida** esperada de la **clase esperada respecto a otros** valores de clase que comparten el valor de los atributos
D2	¿clase real=clase predicha?	**probabilidad** de **estimar** una clase **no óptima**	**probabilidad** de tener una **clase distinta a la esperada** dados los valores de los atributos

Analizado de qué dependen los términos de la 4.3, se observan dos diferencias fundamentales:

- En la divisiones de Breiman y de Domingos, todos los términos están expresados en función de las clases que maximizan la probabilidad, M_r y M_e, mientras que Kohavi calcula cada componente como un sumatorio en todas las clases. En [121] puede consultarse un análisis más detallado de esta diferencia, en el que se hace uso del concepto de *tendencia central*, la clase que tiene más probabilidad de ser predicha por el clasificador (lo que se ha denotado anteriormente por M_e). Para métodos centrados en la *tendencia central*, las definiciones de sesgo y varianza son mejores cuando lo que se quiere es tener unas medidas de cuán fiable es la clase estimada por el clasificador. Sin embargo, definiciones como la de Kohavi que no usan la *tendencia central* son más adecuadas para analizar el rendimiento de un clasificador en general.

- La idea de varianza en el análisis de Breiman es muy diferente a la de los estudios de Kohavi y Domingos, ya que no solamente depende de la estimación, sino también de los propios datos a analizar. Los conceptos de ruido y sesgo sí que son similares, sin embargo, dependiendo únicamente de los datos la primer componente y de una diferencia entre la estimación y el valor real el segundo.

Para poder observar de forma más intuitiva las diferencias entre descomposiciones, se toma a continuación un ejemplo numérico sencillo. Supongamos una máquina en la que se pretende clasificar los tipos de fallos de su funcionamiento en "Eléctrico" (ELE) y "Mecánico" (MEC), mediante dos sensores, S1 y S2, que pueden estar activados (A) o desactivados (D). Como datos de partida para el cálculo, se toma una combinación única de los atributos, los dos sensores activados, según muestra el Cuadro 4.5. En la notación que se ha usado anteriormente, los atributos, x, son los sensores, $x = (S1, S2) = (A, A)$, y los valores reales de la clases para esos atributos, $C_r(x)$, son tanto un fallo eléctrico como mecánico, $C_r(x) = \{ELE, MEC\}$. Las probabilidades reales de cada clase dados los atributos son del 75 % para los fallos eléctricos y del 25 % para los mecánicos, $p_r(ELE|x) = 0{,}75$, $p_r(MEC|x) = 0{,}25$. Por tanto, la clase real más probable es un fallo eléctrico, $M_r(x) = ELE$. Entonces los cálculos del ruido asociado a la combinación de atributos mostrada en el cuadro según la descomposiciones de Kohavi y Breiman-Domingos con pérdida 0/1 son los indicados en 4.18.

Cuadro 4.5: Atributos de ejemplo para cálculo de ruido

S1	S2	Fallo
A	A	ELE
A	A	ELE
A	A	ELE
A	A	MEC

$$\begin{aligned} ruido &= 1 - p_r(M_r|x) = 1 - 0{,}75 = 0{,}25 \quad \text{Breiman, Domingos con L 0/1} \\ ruido &= \tfrac{1}{2}\left[1 - \textstyle\sum_{\text{clases}} p_r^2(x)\right] = \tfrac{1}{2}\left[1 - 0{,}75^2 - 0{,}25^2)\right] = 0{,}1875 \quad \text{Kohavi} \end{aligned} \tag{4.18}$$

Las definiciones de error de Breiman y de Domingos con la pérdida 0/1 son más intuitivas y permiten ver la incertidumbre que hay sobre la clase más probable. Así, en el ejemplo se puede decir que cuando los dos sensores están activos lo más probable es que la clase sea un fallo eléctrico, y que hay una incertidumbre en esa suposición del 25 %. De hecho, si se toman los datos mostrados como conjunto de test y como clasificador aquel que clasifique siempre a la clase más probable para los atributos (fallo eléctrico en este caso), su porcentaje de instancias incorrectamente clasificadas sería justamente el 25 %. Sin embargo, clasificar correctamente las instancias no implica necesariamente haber construido un buen sistema de predicción. Se puede pensar nuevamente en el clasificador que predice siempre la clase mayoritaria, pero en términos de sus estimaciones de probabilidad. Es posible predecir correctamente una clase con una mala estimación de la probabilidad. Tal es el caso en los datos del ejemplo si se predice un fallo eléctrico con total certeza. Entonces, la estimación de probabilidades será del 100 % para los fallos eléctricos y del 0 % para los mecánicos, $p_e = [1\ \ 0]$, cuando la probabilidad real de las clases para los atributos es del 75 % y 25 % respectivamente, $p_r = [0{,}75\ \ 0{,}25]$.

En este contexto, la descomposición de Kohavi se centra en todo el vector de probabilidades y no solamente en la tendencia central. El ruido en este caso estima cuan diferente es el vector de probabilidades real de aquel que permitiría una clasificación perfecta, es decir, de un vector en el que una clase tuviera probabilidad del 100 % y el resto 0 %. En el ejemplo, el error en la definición de Kohavi es un indicador de la distancia entre los vectores $p_r = [0{,}75\ \ 0{,}25]$ y $[1\ 0]$.

4.2. Corrección del sesgo y la varianza a través de ensembles

En la gráfica 3.6 se explicó que había que buscar un punto medio a la hora de elegir la complejidad del modelo, ya que modelos poco complejos llevaban a una falta de ajuste y modelos excesivamente complejos a un sobreajuste. Existe una correspondencia entre estas situaciones y los componentes del error sesgo y varianza. Así, cuando un predictor tiene falta de ajuste su error se debe principalmente al sesgo, mientras que si tiene falta de ajuste la componente predominante en el error es la varianza. Entonces, la obtención de un modelo para el cual la varianza sea muy pequeña será a costa de un sesgo grande, y viceversa. El mínimo error se consigue buscando un punto intermedio donde la suma de ambos componentes sea mínima. Es lo que se conoce en la literatura como *compensación entre el sesgo y la varianza* (*bias-variance tradeoff*, en la bibliografía inglesa).

No siendo posible con un único modelo de predicción disminuir simultáneamente el sesgo y la varianza, surgen los *ensembles*, combinaciones de varios modelos de predicción (llamados *predictores base*) a través de las cuales se busca reducir una componente sin provocar un aumento de la otra.

Los mecanismos de reducción de error de cada una de las técnicas de predicción de tipo ensembles que se han empleado en el trabajo de investigación de la presente Tesis Doctoral no son totalmente diferentes entre sí, sino que en muchos casos comparten estrategias, tomando ciertas variaciones de las mismas. Por ello, se han dividido las técnicas de reducción de error de dichos ensembles en tres categorías:

- Reducción de la varianza por promediado.
- Reducción de la varianza por simplificación de los predictores base.
- Reducción del sesgo por estrategias de aprendizaje secuencial.

A continuación se describen cada una de las tres estrategias, indicando qué ensembles concretos utilizan dichas técnicas, así como algunos trabajos en los que se analice la reducción del error de predicción. Posteriormente, se indican los ensembles en los que se combinan varias de las técnicas de reducción de error.

4.2.1. Reducción de la varianza por promediado

Los mayores representantes de esta técnica de reducción del error son los ensembles tipo Bagging, si bien es una idea usada en más tipos de ensembles, como se indicará posteriormente. El principio estadístico en el que se fundamenta esta técnica de reducción del error se expresa a través de la ecuación 4.19. Dadas N serie de variables aleatorias independientes, X_i, con idéntica varianza, la varianza de su promedio es N veces más pequeña que la de cada una de ellas.

$$Var(\bar{X}) = \frac{Var(X_i)}{N} \tag{4.19}$$

Para obtener varios modelos de predicción a partir de los datos iniciales, cada predictor base se entrena con un conjunto de datos obtenido con muestreo aleatorio *con reemplazamiento* [28] (i.e., una instancia puede aparecer repetida varias veces o no ser considerada para ningún predictor base). En la práctica, la reducción de la varianza se reduce en un factor menor que N, debido principalmente a la correlación entre los modelos obtenidos, pero también al hecho de que modelos entrenados con menos instancias pueden tener más varianza [36].

En el trabajo de Bauer y Kohavi se muestra de forma empírica el efecto de reducción de varianza mediante Bagging [12]. Para que Bagging mejore la precisión de los clasificadores base, es necesario que la técnica de construcción de los mismos sea inestable. Es decir, que pequeños cambios en el conjunto de datos provoquen grandes variaciones en el modelo. En este caso, se consigue reducir la componente del error debida a la varianza dejando prácticamente inalterada la parte correspondiente al sesgo. Sin embargo, con técnicas estables puede empeorar el rendimiento de los modelos base.

4.2.2. Reducción de la varianza por simplificación de los predictores base

En esta técnica de reducción del error de predicción se encuadra el ensemble Random Subspaces. En el análisis de la técnica de predicción, Ho explica que pretende solucionar el problema de sobreajuste que en ocasiones se produce con ensembles cuyos predictores base son árboles de decisión [86]. La situación de falta de ajuste que se indicó en la Figu-

ra 3.6 se corresponde con un error de predicción en la que la varianza es el componente predominante, a cusa de una excesiva complejidad del modelo. La simplificación del modelo que se propone en Random Subspaces consiste en entrenar cada predictor base en subconjunto de los atributos del problema, tomado aleatoriamente.

4.2.3. Reducción del sesgo por estrategias de aprendizaje secuencial

Los ensembles que siguen esta técnica de reducción de error son los de tipo *Boosting*. El tipo de aprendizaje de Boosting está diseñado para reducir el sesgo de un predictor base construyendo un ensemble con un aprendizaje secuencial a partir del mismo, de forma que con cada nuevo predictor base se entrena para corregir el sesgo que resulta de las predicciones anteriores, prestando mayor atención a las instancias donde éstos tuvieron más error.

El efecto que tiene esta técnica de aprendizaje en la varianza, sin embargo, es objeto de debate [34]. Así, algunos autores afirman que Boosting reduce el sesgo en las primeras iteraciones, mientras que en iteraciones posteriores puede reducir además la varianza [59]. Sin embargo, para otros autores Boosting puede sobreajustar los datos, aumentándose la varianza. Por ejemplo, en [124] se da un ejemplo de conjunto de datos para los que Boosting aumenta la varianza. Por tanto, Boosting puede aumentar o reducir la varianza, dependiendo del número de iteraciones y del grado de ajuste del predictor base a los datos considerados. Sin embargo, lo más frecuente es que la varianza aumente aunque a una velocidad mucho más pequeña que la reducción de sesgo [34], lo que se conoce como *tendencia lenta al sobreajuste* (*slow overfitting behaviour*, en la bibliografía inglesa [34]). En [35] se analiza cómo la resistencia de AdaBoost al sobreajuste depende de la función de pérdida con la que se mida el error. Para el caso general de clasificación (pérdida 0/1) el algoritmo tiene mayor resistencia que para otra pérdidas con funciones sustitutas o para pérdidas típicas en regresión como el error cuadrático.

4.2.4. Ensembles que combinan varias estrategias de reducción de error

De los seis tipos de ensembles usados en el estudio de la presente Tesis Doctoral, no se han incluido dos en las tres categorías anteriores: Rotation Forest, Regresión Aditiva

e Iterated Bagging. Los dos últimos ensembles son híbridos entre Bagging y Boosting, por lo que la reducción de error en ellos se explica por los mecanismos mencionados anteriormente (el promediado reduce la varianza y el aprendizaje secuencial el sesgo).

Algunos estudios muestran que Rotation Forest es capaz de reducir tanto el sesgo como la varianza. En los experimentos realizados en [196] se encontró que tanto Rotation Forest como AdaBoost reducían ambas componentes de error, si bien Rotation Forest conseguía una mayor reducción de varianza y AdaBoost una mayor reducción de sesgo. Mientras que la reducción de varianza en Rotation Forest puede explicarse a partir del promediado, como en Bagging, no se ha fundamentado de forma teórica la relación entre la reducción de sesgo y la proyección de los datos en subespacios rotados.

4.3. Teoría del margen y pérdidas basadas en el margen

Un sistema de predicción puede proporcionar resultados muy precisos sobre los propios datos de entrenamiento pero no generalizar bien sobre otros datos. La idea de margen surgió al observar esta situación en problemas de clasificación. Minimizar el porcentaje de instancias incorrectamente clasificadas en el propio entrenamiento no era un buen sistema, sino que era necesario definir algún parámetro que cuantificara el nivel de confianza que se puede tener en una frontera de decisión [178]. Para cada instancia se define el margen como la distancia ente dicha instancia y la frontera de decisión. Entonces, una frontera será más fiable cuanto más margen haya entre todas las instancias y la misma.

Para el caso de clasificación binaria, es habitual representar las clases a partir de los valores $+1$ y -1. De esta forma, es posible construir un clasificador a partir del signo de una frontera de decisión en los atributos, $g(x)$. Así, la clase predicha es $+1$ si el signo de $g(x)$ es positivo, mientras que es -1 si el signo es negativo, correspondiéndose la frontera de decisión con aquellos valores en los que $g(x) = 0$. Entonces, puede definirse el margen, m, a partir del producto del valor real de la clase, y, y del valor de la función de la que se toma el signo para definir la frontera la decisión, $g(x)$, como se muestra en la ecuación 4.20. Por lo tanto, el margen es positivo para una instancia correctamente clasificada, negativo para una instancia incorrectamente clasificada y cero para una instancia en la frontera.

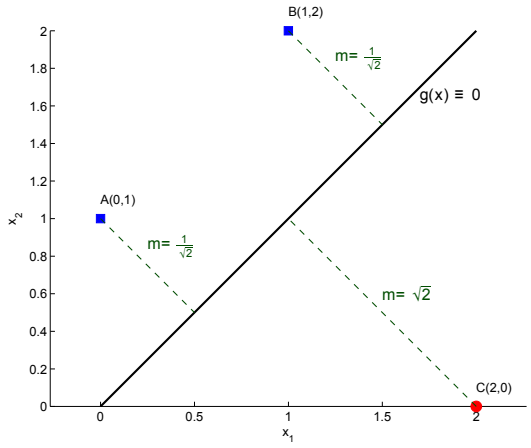


Figura 4.4: Ejemplo numérico de márgenes para un SVM

$$m = y \cdot g(x) \tag{4.20}$$

Geométricamente, el margen es la distancia de cada instancia a la frontera de decisión, como se muestra en la Figura 4.4 para un problema con dos atributos x_1 y x_2. Los cuadrados azules son instancias de la clase negativa, $y = -1$, y el círculo rojos una instancias de la clase positiva, $y = 1$. La frontera es la recta indicada en negro en la figura, que forma 45º con la horizontal ($x_1 = x_2$). Analíticamente, un clasificador SVM definiría la frontera como $g(x) = \frac{1}{\sqrt{2}}(x_2 - x_1)$. Los márgenes de las tres instancias mostradas, A, B y C, se indican en la ecuación 4.21.

$$\begin{aligned} g(A) &= \tfrac{1}{\sqrt{2}}(0-1) = -\tfrac{1}{\sqrt{2}} \rightarrow m(A) = \left(-\tfrac{1}{\sqrt{2}}\right)(-1) = \tfrac{1}{\sqrt{2}} \\ g(B) &= \tfrac{1}{\sqrt{2}}(1-2) = -\tfrac{1}{\sqrt{2}} \rightarrow m(B) = \left(-\tfrac{1}{\sqrt{2}}\right)(-1) = \tfrac{1}{\sqrt{2}} \\ g(C) &= \tfrac{1}{\sqrt{2}}(2-0) = \sqrt{2} \rightarrow m(C) = \sqrt{2}(1) = \sqrt{2} \end{aligned} \tag{4.21}$$

Es posible incluir el margen dentro del concepto de función de pérdida, l, visto en el Apartado 3.4.2, y que servía para expresar el error cometido por un predictor. Cuando se desea obtener un clasificador que maximice el margen, de forma más general puede definirse una función de pérdida cuyo coste sea mínimo para un valor del margen $m = \infty$. Esto se hace a través de las funciones sustitutas (ya introducidas al explicar el ensemble Adaboost en el Apartado 3.5.5), que se expresan como una función del margen. Por ejemplo, para la formulación inicial de AdaBoost para clasificación (AdaBoost.M1), la pérdida

exponencial en función del margen viene dada por la ecuación 4.22. Para el caso del SVM, la función de pérdida en términos del margen se conoce como *función bisagra* (*hinge loss*, en la bibliografía inglesa), indicada en la ecuación 4.23.

$$ł_{exp} = exp(-m) \tag{4.22}$$

$$ł_{bisagra} = \begin{cases} 1-m & \text{si } m < 1 \\ 0 & \text{si } m \geq 1 \end{cases} \tag{4.23}$$

En este contexto, algunos estudios han explicado los buenos resultados experimentales de AdaBoost en base a su capacidad de maximizar el margen, una idea anteriormente usada para los clasificadores SVM (Apartado 3.5.3). Así, en [154] se compara AdaBoost con SVM desde el punto de vista de maximización de margen:

- Mientras que SVM hacía la maximización en un espacio transformado, a través de las funciones kernel, la maximización de AdaBoost se plantea directamente en el espacio de los atributos.
- Los términos que se denotaron por w (peso de cada función base) para explicar SVM con un aproximador de funciones juegan un papel equivalente a los términos que se denotaron por β en la explicación de AdaBoost (pesos de cada *predictor base* o *predictor débil*). Un vector construido con los valores de w y β sobre sus respectivos espacios de atributos debe tener módulo unitario. Sin embargo, para el caso de los SVMs se utiliza la norma 2, ($||w||_2 = 1$) y para el caso de AdaBoost la norma 1 ($||\beta||_1 = 1$).
- Las soluciones al problema de optimización planteado por AdaBoost son más dispersas que las planteadas por SVM, lo que implica que son necesarios menos predictores base que funciones base necesita SVM.

El trabajo de Rätsch también señala el problema que los valores atípicos pueden ocasionar sobre la minimización del margen. En la fase de entrenamiento el predictor puede ser sensible al ruido y generar modelos excesivamente complejos (modelos con mucha varianza) si intenta que el margen sea pequeño para todas las instancias. Por ello, usa el concepto de *margen blando* (*soft margin*), también empleado por los SVMs, para penalizar los modelos excesivamente complejos. Se introduce un término en la función de coste

a minimizar, lo que se conoce como regularización. En la literatura pueden encontrarse otras soluciones para prevenir que AdaBoost no sobreajuste los datos de entrenamiento [125], como la vista en el método de Regresión Aditiva (3.5.5).

También puede usarse el concepto de margen para problemas de regresión, aunque con un objetivo de optimización contrario al empleado para clasificación. Así, para clasificación se busca maximizar el margen, y para regresión minimizarlo. Para ello, también se introduce el margen en las funciones de pérdida, pero en este caso el mínimo coste se tiene cuando el margen vale $m = 0$. Específicamente, el margen es el valor de los residuos, es decir, la diferencia entre el valor real y el estimado. Por ejemplo, para un SVM de Kernel lineal usado para regresión, se considera que las instancias que estén a una distancia muy pequeña (menor que un valor ε) de la recta predicha no suponen ningún coste, siendo para el resto proporcional a la distancia. En la Figura 4.5 se muestra gráficamente el objetivo de minimización para regresión. Se busca que el mayor número de instancias esté en la franja definida por las dos rectas paralelas a la predicción, indicadas en rojo, aunque se permite que algunas instancias superen esa distancia (como la rodeada por un recuadro en línea discontinua). Entonces se dice que la función de pérdida es ε-*insensitiva* (ecuación 4.24) [57]. Como resumen de las funciones de pérdida vistas y su relación con el margen, se muestran la Figura 4.6 para problemas de clasificación y la Figura 4.7 para el caso de problemas de regresión.

$$L_{\varepsilon-insensitiva} = \begin{cases} 0 & \text{si } |m| \leq \varepsilon \\ |m-\varepsilon| & \text{si } |m| > \varepsilon \end{cases} \tag{4.24}$$

En el presente estudio se han usado dos técnicas de clasificación que han sido analizadas en la literatura por la teoría del margen, AdaBoost.M1 y SVM. Sin embargo, en ambos casos no es válido realizar un estudio en términos de margen directamente sobre un problema multiclase. Por un lado, los SVMs que se han usado son clasificadores binarios (un problema multiclase se resuelve descomponiéndolo en varios binarios, como se explica en los artículos "*Identifying maximum imbalance in datasets for fault diagnosis of gearboxes*" y "*An SVM-Based Solution for Fault Detection in Wind Turbines*", expuestos en el apartado 5.2). Por otro, AdaBoost.M1 sí que es multiclase, pero el problema de minimización que define para el caso multiclase no tiene una interpretación geométrica en términos de margen. El algoritmo se basa en encontrar un mínimo de la pérdida 0/1 (el

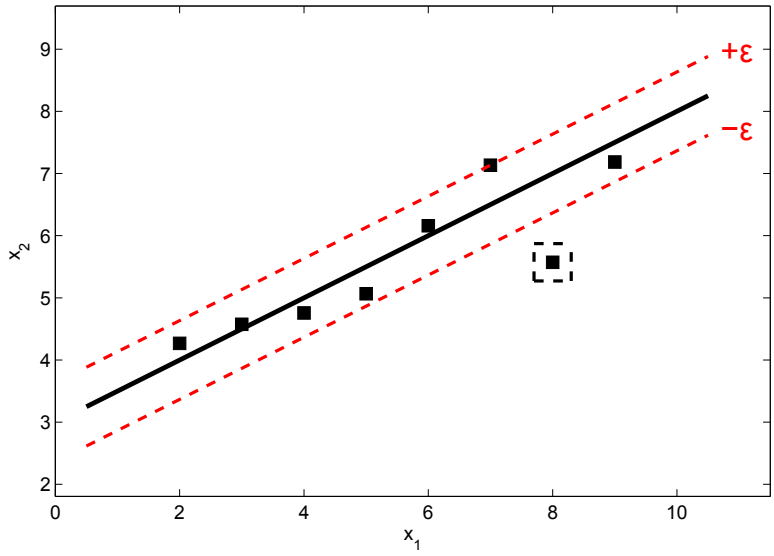


Figura 4.5: SVM para regresión

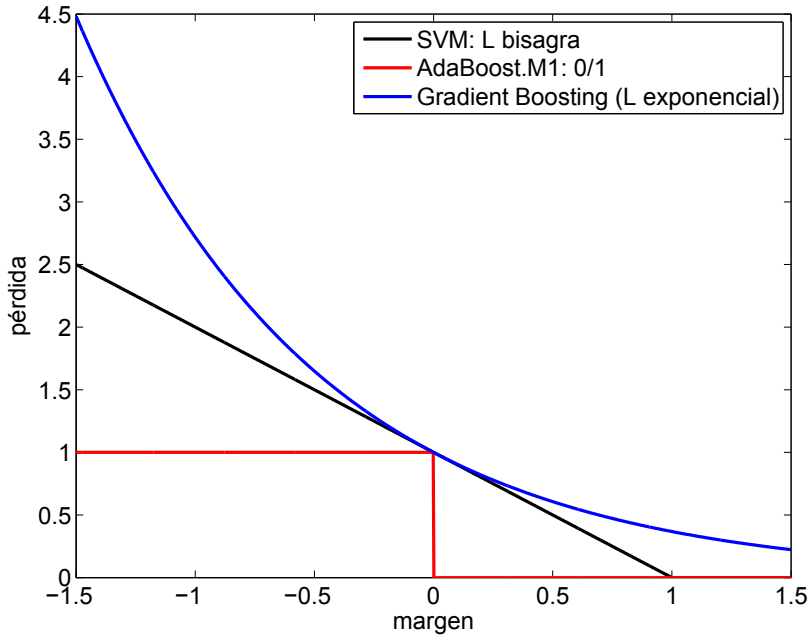


Figura 4.6: Pérdidas en función del margen - Clasificación

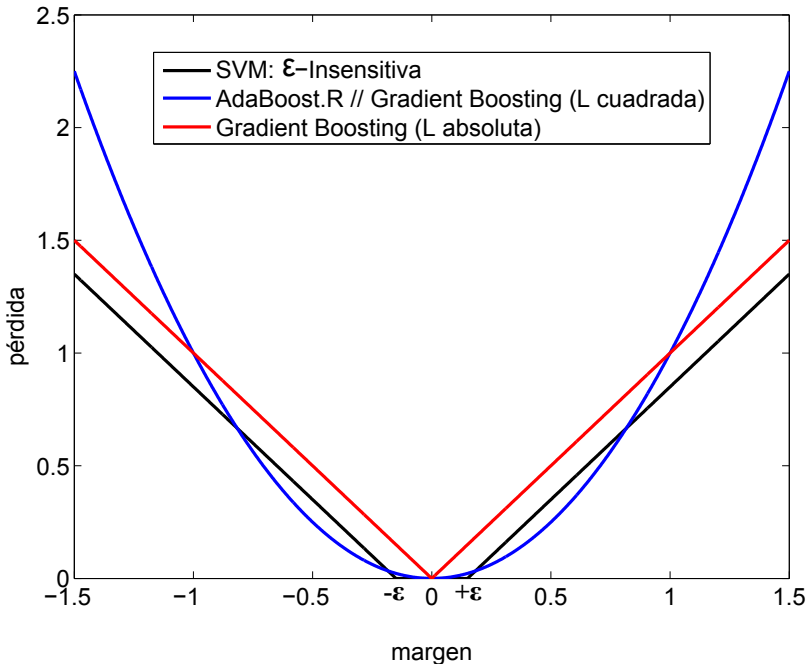


Figura 4.7: Pérdidas en función del margen - Regresión

error de una instancia es 0 si la clase real coincide con la predicha y 1 si no coincide), lo cual puede seguir haciéndose en el caso multiclase, pero sin mantener la idea geométrica. Para dos clases, se podía acotar el error total por la exponencial del margen, m, definido como el producto de la clase, y, por una función escalar, $f(x)$, que variaba entre -1 y 1. Entonces, la clase predicha era simplemente el signo de $f(x)$, lo cual era posible porque las clases se habían definido como $y = 1$ e $y = -1$. Sin embargo, esta formulación tan simplificada no es la que usa para múltiples clases, donde la función con la que trabaja el clasificador es vectorial (las probabilidades de cada clase). La clase predicha en este caso es la que maximiza la probabilidad, es decir, la posición para la cual el vector que devuelve $f(x)$ tiene mayor valor. Por consiguiente, definir $m = yf(x)$ no tiene ningún significado en el caso multiclase.

No se pueden justificar por tanto las técnicas de clasificación multiclase del presente trabajo como una maximización del margen directa sobre el problema planteado como multiclase. Sin embargo, sí que hay estudios que plantean las versiones multiclase de SVM y AdaBoost. En ellos, es necesario definir el margen como una función de salida

escalar que toma como entradas dos variables vectoriales, una que representa el valor real de la clase y otra que representa el valor predicho (y que no tiene por qué coincidir con las probabilidades). Por ejemplo, en [162] se define un AdaBoost multiclase con la función de coste L_M indicada en 4.25, donde $f(x)$ es la salida del predictor, y el vector real de la clase, y^k una representación binaria de la clase k, $< .,. >$ el producto escalar, y C el número de clases.

$$L_M[y, f(x)] = \sum_{k=1}^{C} e^{-\frac{1}{2}\left[<f(x),y> - <f(x),y^k>\right]} \tag{4.25}$$

4.4. Análisis del sesgo y la varianza para un problema industrial

Como ejemplo de descomposición del error en sesgo y varianza se muestra la descomposición resultante para un problema industrial, el fresado de micro-cavidades, expuesto en el artículo "*Modelling laser milling of micro-cavities for the manufacturing of DES with ensembles*" (sección 5.1). De entre todos los regresores analizados en la experimentación, sólo se muestran cinco ensembles con el predictor base para el que se han obtenido mejores resultados (árboles M5 sin poda -M5 U-), así como dicho predictor base y la versión con poda del mismo (M5). Los cinco ensembles considerados son Bagging (BG), AdaBoost.R2 con pérdida lineal (R2-L) y los dos ensembles con estrategias mixtas entre Bagging y Boosting: Regresión Aditiva (AR) e Iterated Bagging (IB). En todos los casos, la componente que más aporta al error es el sesgo, debido a que los modelos han sido entrenados con pocas instancias. Por ello, también se ha añadido el regresor de los k-vecinos, un regresor más propenso a tener una alta componente en varianza [82].

La estimación de las componentes de error se ha realizado siguiendo el esquema de una validación cruzada 10×10, aprovechando de este modo la validación experimental que se utilizó en la publicación correspondiente para medir el error total esperado de los clasificadores, a través de la raíz del error cuadrático medio. De esta forma, se tienen 10 estimaciones para cada instancia (todas las instancias forman parte del conjunto de test 10 veces, una por cada repetición de la validación). Existen en la literatura otras formas de obtener el sesgo y la varianza, basándose en otras validaciones, como el método de retención, explicado en 3.4. Sin embargo, hay estudios que afirman que dichas formas de

estimar la descomposición del error son muy inestables y se pueden usar subconjuntos pequeños para la estimación con diferentes distribuciones de las usadas en la práctica para el aprendizaje de los predictores [199]. Frente a ellos, dichos estudios proponen múltiples validaciones sobre los datos. El esquema seguido en este trabajo para las validaciones múltiples es directamente el de la validación cruzada usada también para estimar el error total. De esta forma, las diferencias entre el error total de predicción de dos predictores pueden ser medidas en diferencias entre componentes.

El problema de estudio considerado tiene la peculiaridad de no tener ruido en los datos. Es decir, cada conjunto de atributos tiene un único valor de la clase asignado. Para cada instancia de los datos de la experimentación se tienen 10 estimaciones del verdadero valor de la clase que es siempre el mismo. Por tanto, el ruido es cero, el sesgo la diferencia entre la media del vector de 10 estimaciones y el verdadero valor de la clase, y la varianza se obtiene con la suma de cuadrados de las diferencias de cada estimación respecto de su media.

Los resultados se muestran en los Cuadros 4.6, 4.7, 4.8 y 4.9 (salidas 1-4, 5-8, 9-12 y 13-14, respectivamente). La primera columna indica el regresor para el que se mide el error, mientras que en la segunda y la tercera se muestran la varianza y el cuadrado del sesgo. Las columnas cuarta y quinta sirven para comparar los componentes de error de los ensembles con los de su regresor base, M5 sin poda. Así, α_1 viene dado por la división del error de varianza del ensemble y error de varianza de su regresor base, mientras que α_2 muestra una división análoga para el caso del sesgo. Por tanto, valores mayores que la unidad indican un aumento y valores menores de la unidad una reducción.

Los resultados de dichos cuadros pueden resumirse de la siguiente forma:

- La componente del error debido al sesgo aporta al menos un 80 % del error total para los cuatro ensembles. También para los k-vecinos, una técnica con la que a priori es más fácil llegar al sobreajuste, el error de predicción se debe fundamentalmente al sesgo.

- Bagging redujo la varianza para todas las salidas. Los otros tres ensembles lo subieron, salvo AdaBoost.R2 e Iterated Bagging para la salida 8 e Iterated Bagging para salida 2 (en la que prácticamente se mantiene estable).

- Para los cuatro ensembles el sesgo o bien bajó o bien se mantuvo prácticamente

estable con una ligera subida (el máximo incremento fue 1.19). Sin embargo, para Bagging sólo en la mitad de las salidas (7) se produjo el decremento del sesgo, frente a las 11 de AdaBoost.R2 y Regresión Aditiva y las 13 de Iterated Bagging. Por tanto, Iterated Bagging fue el método más robusto en lo que a reducción de sesgo se refiere.

- Las proporciones en las que se reduce la varianza con Bagging han sido más llamativas que los factores de reducción del sesgo con los otros tres ensembles. No obstante, al ser el sesgo la componente que más aporta al error en este caso, este hecho no ha conllevado que Bagging proporcione predicciones más precisas.

- El aumento de la varianza con AdaBoost.R2 y Regresión Aditiva es en algunos casos considerable, pasando a valer más del doble. En el caso de la Regresión Aditiva, para 7 de las 14 salidas la varianza se multiplicó por un factor mayor que 4, siendo el máximo valor de dicho factor 7,33, para la salida 14.

En la publicación correspondiente se analizaron los errores cuadráticos medios obtenidos con diferentes técnicas de clasificación, viendo mediante el test *t* remuestreado corregido [137] en qué casos había diferencias estadísticamente significativas, para un nivel de significancia del 95 %. Se resumen los resultados de la siguiente forma:

- Iterated Bagging es el método más robusto ya que no pierde significativamente en ninguna salida.

- La Regresión Aditiva es peor que Iterated Bagging de forma estadísticamente significativa únicamente para la salida 8.

- AdaBoost.R2 (los datos mostrados se corresponden con una función de pérdida lineal) es peor que Iterated Bagging de forma estadísticamente significativa para las salidas 13 y 14.

- Bagging es menos preciso que Iterated Bagging de forma estadísticamente significativa para las salidas 4, 8 y 9.

Cuadro 4.6: Sesgo-Varianza micro-cavidades 1/4

Clasificador	**Varianza**	**Sesgo^2**	α_1	α_2
Salida 1				
M5 U	913598850.31 (2.15 %)	41646929823.37 (97.85 %)	—	—
M5	2149736984.99 (4.52 %)	45400736513.81 (95.48 %)	—	—
R2-L M5 U	1997561409.56 (5.35 %)	35314578591.51 (94.65 %)	2.19	0.85
BG M5 U	385194153.60 (0.92 %)	41401346993.97 (99.08 %)	0.42	0.99
IB M5 U	1490572912.84 (3.56 %)	40402399850.95 (96.44 %)	1.63	0.97
AR M5 U	4151625682.69 (9.98 %)	37447990957.73 (90.02 %)	4.54	0.90
kNN	2223199443.86 (5.52 %)	38050044038.91 (94.48 %)	—	—
Salida 2				
M5 U	32.72 (7.12 %)	426.99 (92.88 %)	—	—
M5	53.37 (9.40 %)	514.27 (90.60 %)	—	—
R2-L M5 U	37.82 (8.12 %)	427.87 (91.88 %)	1.16	1.00
BG M5 U	13.29 (3.20 %)	402.36 (96.80 %)	0.41	0.94
IB M5 U	31.67 (7.52 %)	389.32 (92.48 %)	0.97	0.91
AR M5 U	66.16 (15.80 %)	352.57 (84.20 %)	2.02	0.83
kNN	69.78 (11.85 %)	518.88 (88.15 %)	—	—
Salida 3				
M5 U	1.76 (9.31 %)	17.13 (90.69 %)	—	—
M5	1.47 (7.13 %)	19.18 (92.87 %)	—	—
R2-L M5 U	2.28 (11.83 %)	17.03 (88.17 %)	1.30	0.99
BG M5 U	1.57 (8.33 %)	17.25 (91.67 %)	0.89	1.01
IB M5 U	2.21 (11.99 %)	16.20 (88.01 %)	1.26	0.95
AR M5 U	3.95 (18.91 %)	16.92 (81.09 %)	2.24	0.99
kNN	6.12 (20.17 %)	24.22 (79.83 %)	—	—
Salida 4				
M5 U	1.42 (5.10 %)	26.47 (94.90 %)	—	—
M5	1.80 (5.95 %)	28.38 (94.05 %)	—	—
R2-L M5 U	2.35 (9.22 %)	23.17 (90.78 %)	1.65	0.88
BG M5 U	0.98 (3.41 %)	27.90 (96.59 %)	0.69	1.05
IB M5 U	2.41 (10.27 %)	21.03 (89.73 %)	1.69	0.79
AR M5 U	3.32 (15.47 %)	18.17 (84.53 %)	2.34	0.69
kNN	26.08 (35.01 %)	48.41 (64.99 %)	—	—

Cuadro 4.7: Sesgo-Varianza micro-cavidades 2/4

Clasificador	**Varianza**	**Sesgo^2**	α_1	α_2
Salida 5				
M5 U	5111459.18 (2.01%)	249794104.95 (97.99%)	—	—
M5	4801833.44 (1.76%)	268286880.02 (98.24%)	—	—
R2-L M5 U	13166106.72 (5.36%)	232398288.92 (94.64%)	2.58	0.93
BG M5 U	2413280.26 (1.00%)	239086754.81 (99.00%)	0.47	0.96
IB M5 U	8527937.22 (3.59%)	229035144.54 (96.41%)	1.67	0.92
AR M5 U	24292495.41 (9.22%)	239229415.42 (90.78%)	4.75	0.96
kNN	18883967.01 (7.11%)	246831535.55 (92.89%)	—	—
Salida 6				
M5 U	883638877.26 (2.07%)	41744382105.76 (97.93%)	—	—
M5	2368269292.76 (4.85%)	46458014266.61 (95.15%)	—	—
R2-L M5 U	2347634663.40 (6.39%)	34374038900.63 (93.61%)	2.66	0.82
BG M5 U	384709835.27 (0.92%)	41399425106.37 (99.08%)	0.44	0.99
IB M5 U	1432373683.52 (3.38%)	40956653334.05 (96.62%)	1.62	0.98
AR M5 U	4079360028.56 (9.70%)	37957661608.63 (90.30%)	4.62	0.91
kNN	2218402067.01 (5.51%)	38029399474.32 (94.49%)	—	—
Salida 7				
M5 U	23.26 (5.57%)	394.51 (94.43%)	—	—
M5	14.80 (3.55%)	401.89 (96.45%)	—	—
R2-L M5 U	41.37 (9.20%)	408.22 (90.80%)	1.78	1.03
BG M5 U	12.61 (3.15%)	388.13 (96.85%)	0.54	0.98
IB M5 U	32.07 (8.14%)	361.75 (91.86%)	1.38	0.92
AR M5 U	60.56 (14.66%)	352.68 (85.34%)	2.60	0.89
kNN	85.60 (14.55%)	502.70 (85.45%)	—	—
Salida 8				
M5 U	1.65 (4.66%)	33.79 (95.34%)	—	—
M5	2.91 (7.29%)	36.95 (92.71%)	—	—
R2-L M5 U	1.23 (5.42%)	21.44 (94.58%)	0.74	0.63
BG M5 U	0.37 (1.32%)	27.74 (98.68%)	0.22	0.82
IB M5 U	1.37 (6.95%)	18.38 (93.05%)	0.83	0.54
AR M5 U	2.92 (12.37%)	20.65 (87.63%)	1.76	0.61
kNN	2.03 (7.92%)	23.64 (92.08%)	—	—

Cuadro 4.8: Sesgo-Varianza micro-cavidades 3/4

Clasificador	**Varianza**	**Sesgo^2**	α_1	α_2
Salida 9				
M5 U	1.64 (6.38 %)	24.15 (93.62 %)	—	—
M5	1.52 (5.83 %)	24.61 (94.17 %)	—	—
R2-L M5 U	1.73 (7.42 %)	21.64 (92.58 %)	1.05	0.90
BG M5 U	1.24 (4.67 %)	25.33 (95.33 %)	0.75	1.05
IB M5 U	2.50 (11.48 %)	19.28 (88.52 %)	1.52	0.80
AR M5 U	3.90 (16.24 %)	20.14 (83.76 %)	2.37	0.83
kNN	15.66 (26.14 %)	44.25 (73.86 %)	—	—
Salida 10				
M5 U	3993159.56 (1.70 %)	231238974.42 (98.30 %)	—	—
M5	7522183.26 (2.78 %)	262789970.21 (97.22 %)	—	—
R2-L M5 U	12327475.73 (5.49 %)	212092717.82 (94.51 %)	3.09	0.92
BG M5 U	2070152.23 (0.88 %)	233254600.97 (99.12 %)	0.52	1.01
IB M5 U	7054239.89 (3.09 %)	221109535.38 (96.91 %)	1.77	0.96
AR M5 U	25321173.23 (9.81 %)	232909076.20 (90.19 %)	6.34	1.01
kNN	14948680.80 (6.55 %)	213325899.95 (93.45 %)	—	—
Salida 11				
M5 U	0.00 (2.16 %)	0.08 (97.84 %)	—	—
M5	0.00 (5.16 %)	0.09 (94.84 %)	—	—
R2-L M5 U	0.00 (6.00 %)	0.07 (94.00 %)	2.38	0.82
BG M5 U	0.00 (1.01 %)	0.08 (98.99 %)	0.46	1.00
IB M5 U	0.00 (3.78 %)	0.08 (96.22 %)	1.73	0.97
AR M5 U	0.01 (9.91 %)	0.07 (90.09 %)	4.54	0.91
kNN	0.00 (5.58 %)	0.07 (94.42 %)	—	—
Salida 12				
M5 U	0.01 (4.88 %)	0.11 (95.12 %)	—	—
M5	0.00 (1.70 %)	0.12 (98.30 %)	—	—
R2-L M5 U	0.01 (6.14 %)	0.11 (93.86 %)	1.23	0.97
BG M5 U	0.00 (1.67 %)	0.10 (98.33 %)	0.31	0.94
IB M5 U	0.01 (8.21 %)	0.09 (91.79 %)	1.44	0.83
AR M5 U	0.01 (14.99 %)	0.08 (85.01 %)	2.39	0.70
kNN	0.02 (14.87 %)	0.13 (85.13 %)	—	—

Cuadro 4.9: Sesgo-Varianza micro-cavidades 4/4

Clasificador	**Varianza**	**Sesgo^2**	α_1	α_2
Salida 13				
M5 U	0.00 (2.98 %)	0.00 (97.02 %)	—	—
M5	0.00 (3.75 %)	0.00 (96.25 %)	—	—
R2-L M5 U	0.00 (7.12 %)	0.00 (92.88 %)	2.82	1.13
BG M5 U	0.00 (2.39 %)	0.00 (97.61 %)	0.81	1.02
IB M5 U	0.00 (5.74 %)	0.00 (94.26 %)	1.95	0.98
AR M5 U	0.00 (11.26 %)	0.00 (88.74 %)	4.27	1.03
kNN	0.00 (6.41 %)	0.00 (93.59 %)	—	—
Salida 14				
M5 U	0.00 (1.37 %)	0.01 (98.63 %)	—	—
M5	0.00 (0.89 %)	0.02 (99.11 %)	—	—
R2-L M5 U	0.00 (3.90 %)	0.02 (96.10 %)	3.32	1.13
BG M5 U	0.00 (0.94 %)	0.02 (99.06 %)	0.70	1.03
IB M5 U	0.00 (2.95 %)	0.02 (97.05 %)	2.23	1.01
AR M5 U	0.00 (7.83 %)	0.02 (92.17 %)	7.33	1.19
kNN	0.00 (8.77 %)	0.02 (91.23 %)	—	—

Capítulo 5

Publicaciones

En este capítulo se presentan las publicaciones correspondientes al trabajo de investigación de esta Tesis Doctoral, en la Sección 5.1 las referentes a la fabricación de piezas metálicas de geometría compleja mediante tecnologías láser, y en la Sección 5.2 las correspondientes al diagnóstico de fallos en aerogeneradores.

5.1. Fabricación mediante tecnologías láser

El presente trabajo contiene dos publicaciones referentes a la fabricación mediante tecnologías láser: "*Improvements in modelling of complex manufacturing processes using classification techniques*" y "*Modelling laser milling of micro-cavities for the manufacturing of DES with ensembles*".

En la primera publicación se usan técnicas de clasificación para predecir la rugosidad superficial obtenida con pulido láser de acuerdo con el estándar industrial ISO 4288. Para ello, se analizó un conjunto de datos proporcionado por el Departamento de Ingeniería Mecánica de la Universidad del País Vasco, obtenido mediante la realización de una serie de ensayos con un láser de CO_2 (modelo Rofin Sinar DC 025). Se usaron dos materiales diferentes: una aleación comercial LaserForm^{TM}ST-100 y un acero Orvar Supreme. Mientras que LaserForm ST-100 se usa normalmente para fabricar componentes comerciales mediante SLS, Orvar Supreme se emplea en la producción de moldes. Con el fin de obtener una calidad superficial típica tras operaciones de fresado, se realizó previamente sobre las probetas de ambos metales un fresado con fresa de bola de 12 mm de diámetro

Cuadro 5.1: Atributos considerados para el pulido superficial

Atributo	**Unidades**	**Rango**
Material	Adimensional	1 Orvar, 2 LaserForm
Distancia focal de offset	*mm*	20 - 35
Diámetro del haz	*mm*	1,1-1,9
Potencia del láser	*W*	600 -1600
Tasa de alimentación	mm/min	1100 -1800
Densidad de energía	J/mm^2	10,5 - 81,5
Gas auxiliar	Adimensional	0 Ninguno, 1 Argón, 2 Aire
Rugosidad inicial	μm	1,0 - 6,8

con paso radial variable. Se probaron diferentes densidades de energía, para ajustarse a las condiciones de cada material. Se variaron la tasa de alimentación, potencia del láser y distancia de desenfoque, dado que bajo condiciones industriales la densidad de energía depende de esos tres factores. Tras la realización de los experimentos, se midieron las rugosidades finales y se generó el conjunto de datos para construir los modelos de predicción de Minería de Datos. En total, se probaron 178 condiciones, registrando los atributos mostrados en el Cuadro 5.1. Además de usar los atributos originales registrados en la experimentación, se probó en la experimentación si los clasificadores mejoraban la precisión utilizando una versión discretizada de los atributos originales, así como una versión con atributos binarios indicando si se superaban ciertos niveles en las magnitudes. En total, las cinco versiones de atributos consideradas fueron las siguientes:

- Datos originales recogidos en la experimentación.
- Datos discretizados.
- Atributos binarizados.
- Discretizados + originales.
- Binarizados + originales.

A nivel industrial la calidad de una superficie metálica se mide en términos de su grado de rugosidad superficial. Aunque dicha variable es continua, se clasifica de acuerdo al estándar industrial ISO 4288:1996. Desde el punto de vista de la Minería de Datos existen

dos formas de analizar el problema, según como se trate la clase a predecir. Por un lado, se puede predecir el valor de la rugosidad como variable continua, construyendo un problema de regresión, y posteriormente discretizar dicho valor según el estándar industrial. Por otro lado, puede predecirse directamente el valor discretizado de la rugosidad, mediante un problema de clasificación. Específicamente, el tipo de problema considerado es un caso especial dentro de los problemas de clasificación, conocido como *clasificación ordinal*, en los que existe una relación de orden entre las clases que es deseable que tenga en cuenta el sistema de predicción. Así, si la rugosidad real es de nivel 5, sería más grave predecir que la rugosidad es 3 que predecir que es 4, más cercana al nivel real. Mientras que en un trabajo anterior se analizó la precisión obtenidas mediante técnicas de regresión y posterior discretización, en este estudio se analizaron técnicas de clasificación ordinal.

En el conjunto de datos de la experimentación se registraron niveles de rugosidad del 3 al 7 de acuerdo al estándar ISO 4288. Pero solamente 30 de las 178 instancias pertenecían a las clases 3 y 7, por lo que esas instancias fueron combinadas con los niveles 4 y 6, respectivamente. Esta decisión se llevo a cabo asumiendo que el conjunto de datos no era adecuado para predecir esas condiciones de rugosidad, debiéndose concentrar en niveles intermedios de rugosidad (4-6). De esta forma el conjunto de datos era equilibrado respecto de las clases de rugosidad: 49 instancias para la clase 3-4, 75 instancias para la clase 5 y 54 instancias para la clase 6-7.

Se probaron cinco técnicas de clasificación: un árbol de decisión C4.5, tres ensembles de dicho clasificador base, Bagging, Adaboost y Rotation Forest, y un clasificador SVM. El clasificador más preciso fue un ensemble Rotation Forest con técnicas de clasificación ordinal y utilizando tanto los atributos iniciales como su discretización (83,75 % de acierto). Sin embargo, no se observaron diferencias estadísticamente significativas respecto del peor método de clasificación, SVM con técnicas de clasificación ordinal y los atributos iniciales (77,14 % de acierto).

En la segunda publicación se modela el fresado láser 3D de micro-cavidades. Para ello, se utilizó un conjunto de datos proporcionado por el Departamento de Ingeniería Mecánica y de Construcción Industrial de la Universitat de Girona. Dichos datos se obtuvieron mediante una serie de ensayos realizando dos tipos de micro-cavidades (semiesférica y semicilíndrica) sobre piezas de acero inoxidable 316L, usando un láser de Nd:YAG de 40 de 1064 nm de longitud de onda. Como atributos del problema se consideran los paráme-

Cuadro 5.2: Atributos considerados para el micro-fresado

Variable	Unidades	Rango
Profundidad programada	μm	50 – 90
Radio programado	μm	50 – 83
Longitud programada	μm	0 – 55
Volumen programado	$10^3 \times \mu m^3$	718 – 726
Intensidad	%	60 – 100
Frecuencia	KHz	30 – 60
Velocidad	mm/s	200 – 600
Tiempo	s	9 – 24
MRR programada	$10^3 \times \mu m^3/s$	30 – 81

tros de control de avance del cabezal láser (intensidad, frecuencia, velocidad, tiempo) y los valores programados para las dimensiones a mecanizar y para la tasa de eliminación de material (*material removal rate*, MRR), tal como se indica en el Cuadro 5.2. Por otro lado, como variables de salida del problema (clases de cada predictor), se consideran los valores registrados realmente al realizar las cavidades, tanto de las dimensiones como de la tasa de eliminación de material (Cuadro 5.3).

Se emplearon 39 técnicas de regresión, 32 de ellas ensembles y siete de otro tipo. Las siete técnicas no ensembles empleadas en la experimentación son árboles de regresión RepTree y de modelos M5, regresión lineal, *k*-vecinos, redes neuronales y SVM con kernel lineal y con kernel de base radial. Los 32 ensembles utilizados resultan de combinar 8 tipos de técnicas (Bagging, Iterated Bagging, Adaboost con pérdidas lineal, cuadrada y exponencial, Regresión Aditiva, y Random Subespaces con el 50 % y 75 % de los atributos) con cuatro clasificadores base (árboles RepTree y M5 con y sin poda). La precisión se evaluó midiendo el valor cuadrático medio, RMS (*Root Mean Squared Error*, la raíz cuadrada del Error Cuadrático Medio), en una validación cruzada 10×10 para cada clasificador. Además, se efectuaron test *t* remuestreados corregidos para comparar si las diferencias de precisión entre técnicas eran estadísticamente significativas. La técnica más robusta resulto Iterated Bagging con árboles de decisión M5 sin poda como regresores base, ya que no se obtuvieron resultados significativamente peores que con otra técnica de regresión en ninguna de las 14 salidas consideradas. En la Tabla 5.4 se muestran las precisiones obtenidas para cada una de las salidas predichas, en términos del RMS en unidades de la variable y en porcentaje respecto de la media del valor absoluto de la variable. El grado de

Cuadro 5.3: Variables predichas en el micro-fresado

Variable	Unidades	Rango
Volumen medido	$10^3 \times \mu m^3$	130 – 1701
Profundidad medida	μm	25 – 230,60
Diámetro medido	μm	118,50 – 208.80
Longitud medida	μm	0 – 70,20
MRR medida	$10^3 \times \mu m^3/s$	12 – 121
Error de volumen	$10^3 \times \mu m^3$	-980 – 596
Error de profundidad	μm	-180,60 – 56,90
Error de anchura	μm	-62,80 – -16,63
Error de longitud	μm	-25,50 – 208,80
Error de MRR	$10^3 \times \mu m^3/s$	-61 – 66
Error relativo de volumen	adimensional	-1,36 – 0,82
Error relativo de profundidad	adimensional	-3,61 – 0,63
Error relativo de diámetro	adimensional	-0,55 – 0,10
Error relativo de longitud	adimensional	-0,71 – 0,15

precisión obtenido varía según la salida considerada; la mayor precisión se obtiene para el diámetro medido, con un error de apenas el 2,57 %, mientras que el modelo menos preciso es del error relativo de longitud, con un RMS de hasta el 79,91 %.

Además, se obtuvieron las siguientes conclusiones:

- Los modelos lineales como la regresión lineal y los SVM de kernel lineal no ajustan bien los datos del estudio. Por tanto, son necesarios métodos capaces de operar con relaciones no lineales.

- SVM de kernel de base radial es el único método no ensemble con resultados competitivos, pero necesita ajustar dos parámetros. Las redes neuronales, frecuentemente usadas en problemas de aplicación industrial, no proporcionan buenos resultados en este caso.

- Para algunos tipos de ensembles hay diferencias entre el número de derrotas estadísticamente significativas cuando se usan árboles sin poda o con poda, mientras que dichas diferencias no existen en otros casos; pero en general se obtienen resultados más precisos con árboles sin poda, especialmente para los ensembles con mejores resultados.

Cuadro 5.4: Precisión obtenida para las variables del micro-fresasado

	RMS en unidades de la variable	**RMS en porcentaje (valor absoluto medio)**
Volumen medido	195154,16	38,45
Profundidad medida	19,65	21,82
Diámetro medido	4,30	2,57
Longitud medida	4,78	4,05
MRR medida	14830,83	38,15
Error de volumen	198261,22	64,59
Error de profundidad	19,44	54,11
Error de anchura	4,33	11,20
Error de longitud	4,58	4,77
Error de MRR	14933,58	61,71
Error relativo de volumen	0,28	45,01
Error relativo de profundidad	0,30	9,78
Error relativo de diámetro	0,04	11,46
Error relativo de longitud	0,09	79,91

- Las cinco ensembles más precisos usan como predictor base árboles M5.
- El tipo de pérdida usada en los ensembles Adaboost no parece ser un aspecto decisivo en la precisión.
- Iterated Bagging con regresores base árboles M5 sin poda es el único predictor que no pierde significativamente con otra técnica para alguna de las salidas predichas.

Improvements in modelling of complex manufacturing processes using classification techniques

Pedro Santos, Jesús Maudes, Andrés Bustillo and Juan José Rodríguez

Department of Civil Engineering, University of Burgos, Spain
C/ Francisco de Vitoria s/n, 09006, Burgos, Spain

Abstract. The improvement of certain manufacturing processes often involves the challenge of how to optimize complex and multivariable processes under industrial conditions. Moreover, many of these processes can be treated as regression or classification problems. Although their outputs are in the form of continuous variables, industrial requirements define their discretization in compliance with ISO 4288:1996 Standard. Laser polishing of steel components is an interesting example of such a problem, especially its application to finishing operations in the die and mould industry. The aim of this work is the identification of the most accurate classifier-based method for surface roughness prediction of laser polished components in compliance with the aforementioned industrial standard. Several data mining methods are tested for this task: ensembles of decision trees, classification via regression, and fine-tuned SVMs. These methods are also tested by using variants that take into account the ordinal nature of the class that has to be predicted. Finally, all these methods and variants are applied over different transformations of the dataset. The results of these methods show no significant differences in accuracy, meaning that a simple decision tree can be used for prediction purposes.

Keywords: ensembles, ordinal classification, discretization, process optimization, laser polishing

1 Introduction

The use of high-power lasers in the manufacturing industry has been expanding for over 20 years. The main laser applications for the manufacture of vehicle components in the 1990s involved cutting 2D or 3D metallic sheets. More recently, these applications have not only been used for rapid manufacturing of car components, but also for new tools such as moulds and dies [28].

Over the last ten years, various rapid manufacturing techniques have been developed: Laser Engineered Net Shaping, Selective Laser Sintering (SLS) or Selective Laser Melting. All of these techniques can manufacture fully functional metal parts from raw materials in the form of powder. They achieve significant reductions in manufacturing-time and build parts with very complex geometries that are beyond the scope of traditional manufacturing techniques. Despite these advantages, rapid manufacturing techniques present a fundamental disadvantage for their industrial application: the poor surface quality of the manufactured component [22].

The laser polishing process melts the surface of a component using a laser beam as an energy source. If the process parameters are correctly selected, the laser beam will only melt the peaks of the surface and the melted material will run into the valleys, achieving a smoother topography than the initial one. Experimental results have demonstrated that laser polishing can obtain high-quality roughness surfaces on parts manufactured with SLS that present a very high initial roughness. However, the roughness reduction ratio depends on many variables. Under laboratory conditions, the laser polishing process depends mainly on three factors: the energy density of the laser beam, the surface material, and its initial roughness. Although measurement of the last two factors can be reasonably accurate, the first one is often unknown under real industrial conditions. Furthermore, other process parameters could vary under industrial conditions, such as assistance gas, the beam incidence angle, etc. The influence, under real industrial conditions, of all these parameters has yet to be established, which makes it difficult to optimize the process parameters in the laser polishing process. In consequence, industrial demand for models that take all of these influences into account is high.

Different approaches have been proposed to build a suitable model for this industrial application. Dubey has divided these approaches into three categories: empirical models, analytical models, and artificial intelligence models [8]. This paper falls into the third category. The most common artificial intelligence techniques applied to laser milling and laser polishing include Artificial Neural Networks [2,6], connectionist techniques [3] and particle swarm [6]. To improve the results of a single artificial intelligence model, a combination of two or more models can be built that improves overall model accuracy. These combinations are called ensembles. The prediction capability of an ensemble is built by merging the predictions of the combined models. Ensembles have demonstrated their superiority over single models in many applications such as roughness prediction in milling operations [4], burr detection in the drilling process [10], monitoring of lubricating oil quality [5] and wind turbines fault diagnosis [27]. The artificial intelligence models in this study, have been built using ensembles.

There is another open question related to the artificial intelligence model to be used. Rather than roughness prediction as a continuous variable, the final industrial application will require its accurate prediction in terms of a discretized scale of levels specified in an International Standard, ISO 4288:1996 [16]. Therefore, two methodologies may be followed: i) predict the continuous roughness of the polished component and then discretize the predicted value for each process condition using ISO 4288:1996, or ii) discretize the roughness and train the artificial intelligence model with this new dataset. This open issue has been also studied for other industrial tasks such as roughness prediction in face milling [7], drilling [14] or burr size in drilling [10].

This paper is organized as follows. Section 2 introduces the experimental procedure for data collection and finishes with the dataset description. Section 3 describes the possibilities of ensemble modelling, considering the specific nature of the laser polishing process. Section 4 presents the results of the ensemble modelling and other classification techniques that consider roughness as an ordinal class. Finally, the conclusions and future lines of work are summarized in Section 5.

2 Experimental Procedure and dataset description

The following experimental set-up had the objective of testing a broad range of process parameters used in combination to provide the classifier model with sufficient information on all the relationships between process parameters in the polishing process. A more detailed description of these experiments has previously been published [2].

A CO_2 laser (model Rofin Sinar DC 025) was used to perform all tests. This laser achieves up to 2.5 kW power in continuous mode in an almost Gaussian energy distribution with a 0.4 mm diameter spot. Two different materials were tested for laser polishing: a commercial alloy LaserForm^{TM}ST-100 and Orvar Supreme steel. LaserForm ST-100 is typically used to build up commercial components by SLS. Orvar Supreme is typically used in mould manufacturing. To consider a typical surface, the test blanks were previously milled by a 12mm diameter ball end mill with a variable radial step, in order to obtain a surface quality of a typical milling-operation. The surface topography, in terms of roughness R_a parameter, can be calculated using an equation developed by Quintana [25].

Different energy densities were tested, in order to adjust the conditions to each material. Under industrial conditions [8,6], the energy density depends on three main factors: feed rate of the laser head, power of the laser, and focal offset of the beam. Therefore the experimentation was developed on the basis of a factorial DoE of three factors at three levels, and includes parameter combinations to get results under and over the optimal value presented in the bibliography [21]. The tests consisted of a series of single polishing paths of 20mm in length. Table 1 resumes the parameter values during testing. Three different conditions regarding assistance gas were tested: no assistance gas, air and argon. Altogether, 178 different conditions were tested, the results of which generated a dataset with 178 instances. Finally, the roughness R_a parameter of the laser polished blanks was measured following industrial standard procedure ISO 4287:1997 [17]. Once the experimental measurements of roughness had been made,

Table 1. Process parameter selective for laser polishing tests

	LaserForm^{TM}ST-100			Orvar Supreme		
	Level1	Level 2	Level 3	Level 1	Level 2	Level 3
Power [W]	600	800	1,000	1,200	1,400	1,600
Feed Rate [mm/min]	1,200	1,500	1,800	1,100	1,300	1,500
Focal Offset [mm]	20	27	34	20	27.5	35

the dataset for the artificial intelligence model was generated. The variables included in the dataset and their variation ranges, see Table 2, are: material, focal offset distance, spot diameter, laser power, feed rate, energy density, assistance gas, initial roughness and final roughness. Roughness was discretized according to industrial standard ISO 4288:1996 [16]. Five different levels of roughness were identified in the dataset: from level 3 to level 7 of standard ISO 4288:1996. But only 30 of the 178 instances referred to roughness classes 3 and 7, therefore these instances were combined with classes 4

Table 2. Variables, units and ranges used during the experiments

Variable (Units)	Input / Output	Range
Material	Input variable	1 Orvar, 2 LaserForm
Focal offset distance (mm)	Input variable	20 - 35
Spot diameter (mm)	Input variable	1.1-1.9
Laser Power (W)	Input variable	600 -1,600
Feed rate (mm/min)	Input variable	1,100 -1,800
Energy density (J/mm^2)	Input variable	10.5 - 81.5
Assistance gas	Input variable	0 none, 1 Argon, 2 Air
Initial roughness (μm)	Input variable	1.0 - 6.8
Final roughness (μm)	Output variable	0.2 - 6.2 (continuous) or 4-5-6 (discretized)

and 6, respectively. This decision was taken on the assumption that the dataset would not be suitable to predict process conditions in these outer limits and should concentrate on correctly dividing the middle roughness levels 4-6. Within this dataset the three different levels of roughness, were balanced: 49 (Class 3-4), 75 (Class 5) and 54 instances (Class 6-7).

3 Data Mining Techniques Applied in the Study

As detailed in the previous section, the aim is to predict final roughness. Final roughness was categorized using three ordinal classes which label roughness to smoothness on a descending scale (a roughness level of 4 is smoother than 5, and so on).

Ordinal classification methods assume that an order exists among the classes. Classification of laser polishing is an ordinal class problem, where classes represent a grade of final roughness. There are methods for ordinal classification [11] [19] that exploit the information derived from such orders to arrive at greater accuracy.

The standard approach to ordinal classification probably consists of predicting a continuous variable (i.e., numeric final roughness in this problem), which is then discretized into a categorical set of prediction classes [19]. This standard approach is ideal for our problem, as (i) in [4], experimental tests confirmed that Bagging [1] of regression trees gave better results than other state-of-the-art regressors, and (ii) a discretization method [16] exists to map this numeric variable onto the classes that need to be predicted. Therefore this technique is tested in the study and denoted as *Classification-Via-Regression*.

In [11], the ordinal multiclass problem with k classes is transformed into $k-1$ binary problems. The ith problem tries to predict whether the class is bigger than the ith ordinal class value. Therefore $k-1$ binary classifiers have to be trained, and their probability estimations are used to compute the final prediction.

The main advantage of this approach is that it can combine any kind of binary classifiers to tackle the $k-1$ individual binary sub-problems. For example, in [11] C4.5 decision trees are used, a technique that is labeled OCC (i.e., *Ordinal Class Classifier*) in this work.

As stated in [11,12], an analogous strategy to OCC can be applied to input attributes. Each input attribute a_i can be discretized using a supervised discretization method that maps it into m nominal categories. Each category covers a range for continuous values of a_i, so there are $m-1$ splitting points delimiting the m ranges. Hence, the resulting m categories can be grouped into $m-1$ binary features that signify that a_i is either larger or smaller than one of these splitting points. The discretization method used in [12] was the Fayyad-Irani supervised discretization [9].

The main advantage of constructing these new boolean features is that learning algorithms can be sensitive to the order in the categories obtained by the discretization. Moreover, some algorithms such as decision trees can use the split points given by the discretization method and its global view from the dataset, mapped into a local region formed by the instances belonging to a branch. These combinations of a global and local view from the dataset are useful to reduce classifier overfitting.

As in [12], the original dataset is denoted as (RAW) in this study. The discretized version of the dataset using the Fayyad-Irani method over all numeric attributes is denoted as (DISC), and finally, a discretized dataset where discrete attributes have been transformed into a set of binary attributes representing their ordinal nature is denoted as (ORD). There are another two dataset versions used in the study, according to whether original numeric features are maintained alongside the discretized features. DISC_KEEP and ORD_KEEP are respectively the DISC and ORD versions that retain the original numeric features.

Some of the more representative classification methods were tested. In a first group, two singleton methods are considered:

1. Decision Trees: C4.5 Quinlan decision tree was used [24]. As previously pointed out, decision trees can take advantage of binary features computed on ORD and ORD_KEEP dataset versions. Another two remarkable issues are that decision trees can process directly nominal features, such as those computed in the DISC and DISC_KEEP versions and can also work on multiclass problems.
2. Support Vector Machine (SVM) [29]. Radial Function Basis Function Kernel was used for this classifier. SVM is one of the most popular and successful state-of-the-art classifiers [30]. It reports very competitive results provided that its parameters (i.e., the slack variable and gamma) have been properly tuned. SVM is a binary classifier (i.e., it only can predict on two classes problems). Multiclass problems for SVM are transformed into k binary problems, where k is the number of classes. Each binary problem predicts if an instance belongs to one of the classes vs. the rest (i.e., one vs. all approach). Another interesting method for SVM to tackle with the three classes in the polishing-laser dataset is OCC. OCC also divides the multiclass problem into a few binary problems and takes its ordinal nature into account.

The other group of classifier methods to consider are the ensembles [20]. Ensembles gather predictions from a set of so-called base classifiers. The final predictions are made by simple majority voting, weighted voting, or any other combination schema. Most popular ensembles combine base classifiers trained using the same algorithm. Hence, a fundamental element of ensembles is their strategy that ensures the resulting base classifiers are different. If all base classifiers agree, there would be no difference between

using only one base classifier or an ensemble of base classifiers. This property of ensembles is known as the *diversity* of base classifiers [20]. Diversity may be reached by introducing some kind of perturbation in the version of the dataset that is used to train each base classifier. Hence, base classifiers that are highly sensitive to changes in the training dataset (i.e., *unstable* base classifiers) are appropriate for ensembles. Decision trees are an example of unstable classifiers whereas SVM is an example of a stable one.

In this study we have used the following ensembles, taking C4.5 trees as base classifiers:

1. Bagging [1]: the diversity of this ensemble depends on training each base classifier by sampling with replacement from the original dataset. Bagging reduces the component of the error debt to the casual distribution of the instances forming the training dataset (i.e., variance error component [18]).
2. Boosting [13]: in Boosting, each base classifier is trained by focusing on training instances misclassified by the previous base classifier. A weight is given to each instance using the prediction of the previous base classifier. This weight increases when the instance has been misclassified. Final prediction is made by using a weighted voting schema where the larger training error on each base classifier results in a lower weight. There are a lot of Boosting variants. In this paper, AdaBoost M1 [13] is used because it is the most popular.
3. Rotation Forest [26] trains each base classifier by grouping their attributes into subsets (e.g., subsets of three attributes are usually taken). Then PCA (*Principal Component Analysis*) is computed for each group using a subsample from the training set. The whole dataset is transformed according these projections and can then be used to train a base classifier.

4 Results and Discussion

WEKA [15] that was used for the experimental validation provides implementations for all the methods and data transformations in use. Eight methods were tested: C4.5 decision tree, SVM with Radial Basis Function kernel, Bagging, AdaBoost M1, Rotation Forest, and Classification via Regression applying results in [4].

The default parameters of these methods were used, except that:

1. The size of the ensembles was set at 100. This setting was chosen because in [4] Bagging for regression used 100 regression trees and classification via that regression will be tested, so the rest of ensembles take that size to ensure comparable results.
2. The base classifiers for all the ensembles was a C4.5 decision tree, without pruning.
3. SVM was always tuned for each training data partition in the validation. The slack variable and gamma parameters were tuned.

These methods were tested using the following dataset versions, RAW, DISC, DISC_KEEP, ORD and ORD_KEEP described in the previous section.

All the methods, except for classification via regression, were tested using these dataset versions. The supervised discretization was computed for each training fold.

Finally, because the dataset has an ordinal class, OCC was tested. So, all the configurations were validated with and without OCC.

10×10 cross-validation was used in the experimental validation as in [4]. Table 3 shows the accuracy results for all the methods, except for the classification via regression the accuracy of which was 79.26%. The best accuracy (83.75%), in bold in Table 3,

Table 3. Accuracies reached in the datasets tested

	No OCC RAW	OCC RAW	No OCC DISC	OCC DISC	No OCC DISC_K	OCC DISC_K	No OCC ORD	OCC ORD	No OCC ORD_K	OCC ORD_K	Avg
C4.5	82.73	82.73	82.78	82.73	83.23	82.18	82.62	82.73	82.4	82.68	82.68
SVM	78.27	77.14	82.61	82.18	81.52	81.17	82.23	81.95	81.26	81.24	80.96
Bagging	82.51	82.29	82.67	82.57	82.67	82.12	82.56	82.73	82.23	82.06	82.44
M1	81.33	80.92	82.45	81.17	80.81	80.75	82.16	80.88	81.04	80.69	81.27
Multiboost	81.89	82.5	80.92	80.6	82.45	82.21	80.53	80.94	82.5	82.32	81.69
RotForest	82.17	83.01	82.56	82.17	82.9	**83.75**	81.89	82.23	82.29	82.28	82.53
Average	81.7	81.43	82.44	81.9	82.56	82.03	82.17	81.91	82.06	81.88	

was reached by Rotation Forest using OCC from DISC_KEEP version, and the worse result (77.14%), by SVM using OCC on RAW version. Classification via regression performance (79.26%) is therefore closest to the worse method.

The resampled *t-test* [23], applied between the best and the worse configurations, showed no significant difference (significance 5%). There were therefore no significant differences between the methods. Moreover, a single C4.5 without OCC on the DISC_KEEP version reached 82.73%. The decision tree may be the best classifier option, because it is the fastest and simplest model and because its decision nodes can be interpreted on sight to extract valuable knowledge.

OCC does not increase accuracy, as was expected. This could be because the problem has only three classes. The three classes generate only two binary problems, which are also imbalanced problems.

Average accuracies on SVM, and AdaBoostM1 are worse than average accuracies of the rest of the methods. This might suggest that data is somewhat noisy as these methods are more sensitive to learn the noise. This idea is reinforced by the low average accuracy on data versions without any discretization (i.e., RAW).

Regarding data versions, DISC_KEEP appeared to be the version that yielded the best results. The two top configurations are computed over this version, and the average accuracy of DISC_KEEP was the best. Discretization is used to avoid overfitting and to increase classifiers generalization. Keeping the original attributes gives a broader range of choices in decision trees branching. On the other hand, average accuracy in ORD versions does not increases accuracy as expected.

5 Conclusions and Future lines of Work

This research has presented the results of an investigation to identify the most appropriate methodology to solve a real-life industrial problem concerning the laser polishing of metallic components. This industrial task, like many others, can be treated as a regression or a classification problem, because the industrial requirement is based on an ISO Standard that discretizes the output, even though it is a continuous variable. Several ensembles were investigated to achieve the best practical solution to this interesting problem. The two proposed methodologies were: building a regression model and discretizing its output using the industrial Standard ISO 4288:1996 and discretizing the output variable of the dataset using this Standard and then building a classification model afterwards.

A real dataset generated under real industrial conditions was used to validate this approach. The dataset included 178 instances with 8 input variables and 1 output, the final roughness of the polished blank. The correct evaluation of three different levels of roughness with a balanced number of instances within them were researched. This research shows that there are no significant differences between the machine learning techniques tested. For this reason the simplest method (i.e., ensembles of C4.5 decision trees) is probably the best solution for the problem, as it is faster than the other methods and is the only method that provides insight on the classification criteria. Ordinal classification was treated using OCC but the results did not improve, maybe because of the reduced number of classes. Techniques that preprocess data including some discretization variants were also tested. It appears that discretization slightly helped to increase accuracy. The best results in the experimental validation were reached by merging the discretized attributes and the original features.

Future work will consider other ensemble methods, using ensembles from other methods instead of decision trees and studying the use of non-homogeneous ensemble models. These ensembles are built by combining different methods, (e.g., SVM and Decision Trees) and could improve final model accuracy by specializing each base learner in an area of the problem space. With regard to preprocessing, the use of supervised projections should be tested, as Rotation Forests, a method that uses an internally unsupervised projection, gave some of the best results. A methodology improving accuracy results could also be applied to other industrial problems that predict surface roughness in compliance with Industrial Standard ISO 4288:1996.

Acknowledgments

This investigation has been partially supported by the Projects CENIT-2008-1028, TIN2011-24046, IPT-2011-1265-020000 and DPI2009-06124-E/DPI of the Spanish Ministry of Economy and Competitiveness. This work has been made possible thanks to the support received from University of the Basque Country, which provided the laser polishing data and performed all the experimental tests. The authors would especially like to thank Dr. Aitzol Lamikiz and Dr. Eneko Ukar for their kind-spirited and useful advice.

References

1. Breiman, L.: Heuristics of instability and stabilization in model selection. The annals of statistics 24(6), 2350–2383 (1996)
2. Bustillo, A., Díez-Pastor, J., Quintana, G., García-Osorio, C.: Avoiding neural network fine tuning by using ensemble learning: application to ball-end milling operations. The International Journal of Advanced Manufacturing Technology 57(5), 521–532 (2011)
3. Bustillo, A., Sedano, J., Villar, J., Curiel, L., Corchado, E.: Ai for modelling the laser milling of copper components. Intelligent Data Engineering and Automated Learning–IDEAL 2008 pp. 498–507 (2008)
4. Bustillo, A., Ukar, E., Rodriguez, J., Lamikiz, A.: Modelling of process parameters in laser polishing of steel components using ensembles of regression trees. International Journal of Computer Integrated Manufacturing 24(8), 735–747 (2011)
5. Bustillo, A., Villar, A., Gorritxategi, E., Ferreiro, S., Rodríguez, J.J.: Using ensembles of regression trees to monitor lubricating oil quality. In: Proceedings of the 24th international conference on Industrial engineering and other applications of applied intelligent systems conference on Modern approaches in applied intelligence - Volume Part I. pp. 199–206. IEA/AIE'11, Springer-Verlag, Berlin, Heidelberg (2011), http://dl.acm.org/citation.cfm?id=2025756.2025782
6. Ciurana, J., Arias, G., Ozel, T.: Neural network modeling and particle swarm optimization (pso) of process parameters in pulsed laser micromachining of hardened aisi h13 steel. Materials and Manufacturing Processes 24(3), 358–368 (2009)
7. Díez-Pastor, J., Bustillo, A., Quintana, G., García-Osorio, C.: Boosting projections to improve surface roughness prediction in high-torque milling operations. Soft Computing-A Fusion of Foundations, Methodologies and Applications pp. 1–11 (2012)
8. Dubey, A., Yadava, V.: Laser beam machining—a review. International Journal of Machine Tools and Manufacture 48(6), 609–628 (2008)
9. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning. pp. 1022–1027 (1993)
10. Ferreiro, S., Sierra, B., Irigoien, I., Gorritxategi, E.: Data mining for quality control: Burr detection in the drilling process. Computers & Industrial Engineering 60(4), 801–810 (2011)
11. Frank, E., Hall, M.: A simple approach to ordinal classification. Machine Learning: ECML 2001 pp. 145–156 (2001)
12. Frank, E., Witten, I.: Making better use of global discretization (1999)
13. Freund, Y., Schapire, R., et al.: Experiments with a new boosting algorithm. In: Machine Learning-International Workshop. pp. 148–156. Morgan Kaufmann Publishers, Inc. (1996)
14. Grzenda, M., Bustillo, A., Zawistowski, P.: A soft computing system using intelligent imputation strategies for roughness prediction in deep drilling. Journal of Intelligent Manufacturing pp. 1–11 (2012)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.: The weka data mining software: an update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)
16. International Organization for Standardization: ISO-4288. Geometrical Product Specifications (GPS): Rules and procedures for the assessment of surface texture (1996)
17. International Organization for Standardization: ISO-4287. Geometrical Product Specifications (GPS) — Surface texture: Profile method — Terms, definitions and surface texture parameters (1997)
18. Kohavi, R., Wolpert, D., et al.: Bias plus variance decomposition for zero-one loss functions. In: Machine Learning-International Workshop. pp. 275–283. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1996)

19. Kramer, S., Widmer, G., Pfahringer, B., Groeve, M.: Prediction of ordinal classes using regression trees. Fundamenta Informaticae 47(1-2), 1–13 (2001)
20. Kuncheva, L.: Combining pattern classifiers: methods and algorithms. Wiley-Interscience (2004), http://books.google.es/books?id=9TJ6igZtqWAC
21. Lamikiz, A., Sanchez, J., Lopez de Lacalle, L., Arana, J.: Laser polishing of parts built up by selective laser sintering. International Journal of Machine Tools and Manufacture 47(12), 2040–2050 (2007)
22. Lü, L., Fuh, J., Wong, Y.: Laser-induced materials and processes for rapid prototyping. Springer (2001)
23. Nadeau, C., Bengio, Y.: Inference for the generalization error. Machine Learning 52(3), 239–281 (2003)
24. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
25. Quintana, G., De Ciurana, J., Ribatallada, J.: Surface roughness generation and material removal rate in ball end milling operations. Materials and Manufacturing Processes 25(6), 386–398 (2010)
26. Rodríguez, J., Kuncheva, L., Alonso, C.: Rotation forest: A new classifier ensemble method. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28(10), 1619–1630 (oct 2006)
27. Santos, P., Villa, L., Reñones, A., Bustillo, A., Maudes, J.: Wind turbines fault diagnosis using ensemble classifiers. Advances in Data Mining. Applications and Theoretical Aspects 7377, 67–76 (2012)
28. Tuck, C., Hague, R., Ruffo, M., Ransley, M., Adams, P.: Rapid manufacturing facilitated customization. International Journal of Computer Integrated Manufacturing 21(3), 245–258 (2008)
29. Vapnik, V.: The nature of statistical learning theory. Springer (1999)
30. Wu, X., Kumar, V.: The top ten algorithms in data mining, vol. 9. Chapman & Hall/CRC (2009)

Modelling laser milling of micro-cavities for the manufacturing of DES with ensembles

Pedro Santos[1], Daniel Teixidor[2], Jesús Maudes[1] and Joaquim Ciurana[2]

[1] Department of Civil Engineering, University of Burgos, Spain
[2] Department of Mechanical Engineering and Industrial Construction, Universitat de Girona, Spain

A set of designed experiments, performed with a pulsed Nd:YAG laser system, using 316L Stainless Steel as the working material, serve to study the laser milling process of micro-cavities in the manufacture of Drug-Eluting Stents (DES). Diameter, depth and volume error are considered to be optimized as functions of the process parameters, which include laser intensity, pulse frequency and scanning speed. Two different DES shapes are studied that combine semi-spheres and cylinders. Process inputs and outputs were defined by considering the process parameters that can be changed under industrial conditions and the industrial requirements of this manufacturing process. In total, 162 different conditions were tested. This process was modeled with state-of-the-art data-mining regression techniques: Support Vector Regression, Ensembles, Artificial Neural Networks, Linear Regression and Nearest Neighbor Regression. Ensemble regression appears to be the most suitable technique to study this industrial problem. Specifically, Iterated Bagging ensembles with unpruned model trees outperformed the other methods in the tests. This method can predict the geometrical dimensions of the machined microcavities with relative errors related to the main average value in the range of 3 to 23%, which are considered very accurate predictions, in view of the characteristics of this innovative industrial task.

1 Introduction

Laser-milling technology has become a viable alternative to conventional methods for producing complex micro features on difficult-to-process materials. It is increasingly employed in the industry, because of its established advantages [1]. As a non-contact material-removal process, laser machining removes smaller and more precise amounts of material, applies highly localized heat inputs to the workpiece, minimizes distortion, involves no tool wear, and is not subject to certain constraints such as maximum tool force, buildup edge formation and tool chatter. Micro-manufacturing processes in the field

of electronics and medical and biological applications have become a growing area of research. High-resolution components, high-precision and small feature size are needed in this field, as well as real 3D fabrication. Thus, the use of laser machining to produce medical applications has become a growing area of research, one example of which is the fabrication of coronary stents. This research looks at the fabrication and performance of the DES. Some of these DES are metallic stents that include reservoirs that contain the polymer and the drug [2], such as the Janus TES stent [3] which incorporates micro-reservoirs cut into its abluminal side that are loaded with the drug. The selection of the laser system and the process parameters significantly affects the quality of the micro-feature that is milled and the productivity of the process. Although there are several studies which deal with the effect of the process parameters on the quality of the final parts of laser milling, few of them study this effect on a micro scale. Many experimental research works have studied the influence of scanning speed, pulse intensity and pulse frequency on the quality and productivity of the laser milling in different materials on a macro scale [4, 5, 6, 7]. There are many works on micro-scale machining that have investigated laser-machining processes in laser micro-drilling [8, 9, 10], laser micro-cutting [11, 12, 13] and laser micro-milling in 2D [14, 15]. However, there is little research on laser 3D micro-milling. Pfeiffer et al. [16] studied the effects of laser process parameters on the ablation behaviour of tungsten carbide hard metal and steel using a femtosecond laser for the generation of complex 3D microstructures. Karanakis et al. [17] demonstrated the laser milling capacity of a picoseconds laser in different materials (stainless steel, alumina and fused silica). Surface topology information was correlated to incident power density, in order to identify optimum processing. Qi et al. [18] used a fiber laser to machine complex shapes. They developed a thermal ablation model to determine the ablated material volume and the dimensions and optimized the parameters to achieve maximum efficiency and minimum thermal effects. Finally, Teixidor et al. [19] studied the effects of scanning speed, pulse intensity and pulse frequency on target width and depth dimensions and surface roughness for the laser milling of micro-channels on tool steel. They presented a second-order model and a multi-objective process optimization to predict the responses and to find the optimum combinations of process parameters.

Although the manufacturing industry is interested in laser micro-milling and some research has been done to understand the main physical and industrial parameters that define the performance of this process, the conclusions show that analytical approaches are necessary for each real case due to their complexity. Artificial intelligence approaches represent a suitable alternative to such tasks due to their capacity to deal with multivariate processes and experimental uncertainties. Many artificial intelligence techniques have been applied to macro-scale milling [20, 21, 22], but there are few examples of the application of such techniques to laser micro-milling [23]. Artificial Neural Networks (ANNs) have been proposed to predict the pulse energy for a desired

depth and diameter in micro-milling [24], the Material Removal Rate (MRR) for a fixed ablation depth depending on scanning velocity and pulse frequency [25] and cut kerf quality, in terms of dross adherence during non-vertical laser cutting of 1 mm thick mild steel sheets [26]. Finally, regression trees have been proposed to optimize geometrical dimensions in micro-manufacturing of micro-channels [27]. Neither of these studies used ensembles for process modeling, a learning paradigm in which multiple learners (or regressors) are combined to solve a problem. A regressor ensemble can significantly improve the generalization ability of a single regressor and can provide better results than an individual regressor in many applications [28, 29, 30]. Ensembles have demonstrated their suitability for modeling macro-scale milling and drilling [31, 32, 33, 34], especially because they can achieve highly accurate prediction with lower tuning time of the model parameters [33]. In view of the lack of research in the literature on the modeling of 3D micro-geometries by laser milling using ensembles, the objective of this work is to study the capability of these data-mining techniques to model this industrial task, depending on the different inputs and outputs that can be considered priorities from an industrial point of view.

2 Experimental procedure and data collection

The experimental apparatus described in this study gathered the data needed to create the models. The experimentation consisted of milling micro-cavities in a 316L Stainless Steel workpiece using a laser system. A Deckel Maho Nd:YAG Lasertec 40 machine with a 1,064 nm wavelength was used to perform the experiments. The system is a lamp-pumped solid-state laser that provides an ideal maximum pulse intensity of 1.4 W/cm^2 (theoretically estimated as [14]), due to the 100 W average power and 30 μm beam spot diameter. The SS316L workpiece material was selected because it is a biocompatible material commonly used in biomedical applications and specifically for the fabrication of coronary stents. Two different geometries were used for the experiments. The first geometry consisted of a half-spherical shape defined by depth and diameter dimensions. The second geometry was a half-cylindrical shape with a quarter sphere on both sides, defined by depth, diameter and length dimensions. Both geometries are presented in the Figure 1. At the same time, the geometries were fabricated with different combinations of dimensions keeping the same volume. Table 1 and Table 2 respectively present the three combinations of dimensions for both the spherical and the cylindrical geometries. These geometries and dimensions were selected, because they provide sufficient space to machine the cavities in these cardiovascular drug-eluting stent struts, an important part of their manufacturing process.

A full factorial design of experiments was developed, in order to analyze the effects of Pulse Frequency (PF), Scanning Speed (SS) and Pulse Intensity levels (PI, percentage of the ideal maximum pulse intensity) on the responses.

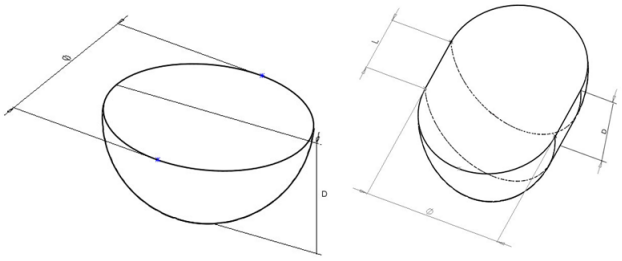


Fig. 1. Cavity geometries used in the experiments

Table 1. Sphere geometry dimensions

Geometry	Depth (μm)	ϕ (μm)	Volume (μm^3)
Sphere 1 (e1)	50	166	721414
Sphere 2 (e2)	70	140	718377
Sphere 3 (e3)	90	124	724576

Table 2. Cylinder geometry dimensions

Geometry	Depth (μm)	ϕ (μm)	Length (μm)	Volume (μm^3)
Cylinder 1 (c1)	50	130	55	723220
Cylinder 2 (c2)	70	110	46	721676
Cylinder 3 (c3)	90	100	36	725707

Some screening experiments were performed to determine the proper parameters levels. Three different levels were selected from the results for each input factor, which are presented in Table 3. This design of experiments resulted in a total of 162 experiments; 27 combinations for each geometry under study. All the experiments were machined in the same 316L SS blank under the same ambient conditions. The response variables under investigation were the cavity dimensions (depth and radium) and the volume of removed material. A confocal microscope Axio CSM 700 from Carl Zeiss was used for the dimensional measurements and for characterization of the cavities. Moreover, negatives of some of the samples were obtained with surface replicant silicone, in order to obtain 3D SEM images.

Table 3. Factors and factor levels

Factors	Factor Levels		
Scanning Speed (SS) [mm/s]	200	400	600
Pulse Intensity (PI) [%]	60	78	100
Pulse Frequency (PF) [kHz]	30	45	60

Table 4. Input Variables

	Variable	Units	Range	Relationship
$\mathbf{x_1}$	*Programmed Depth*	μm	50 – 90	Independent
$\mathbf{x_2}$	*Programmed Radio*	μm	50 – 83	Independent
$\mathbf{x_3}$	*Programmed Length*	μm	0 – 55	Independent
$\mathbf{x_4}$	*Programmed Volume*	$10^3 \times \mu m^3$	718 – 726	$4/3\pi x_2^3 + 1/2 x_2^2 x_3$
$\mathbf{x_5}$	*Intensity*	%	60 – 100	Independent
$\mathbf{x_6}$	*Frequency*	KHz	30 – 60	Independent
$\mathbf{x_7}$	*Speed*	mm/s	200 – 600	Independent
$\mathbf{x_8}$	*Time*	s	9 – 24	Independent
$\mathbf{x_9}$	*Programmed MRR*	$10^3 \times \mu m^3/s$	30 – 81	x_4/x_8

Having performed the experimental tests, the inputs and outputs for the datasets had to be defined, to generate the data sets for the data-mining modeling. On the whole, the selection of the inputs is easy, because they are set by the specifications of the equipment: the inputs are the parameters that the process engineer can change in the machine. They are the same as those considered to define the experimental tests explained above. Table 4 summarizes all the selected inputs, their units, ranges and the relationship that they have with other inputs.

Regarding the data-set outputs, their definition takes into account the different interests that the industrial manufacturing of DES can have. In some cases, a productivity orientation will encourage the process engineer to optimize productivity (in terms of the MRR) keeping geometrical accuracy under certain acceptable thresholds (by fixing a maximum relative error in geometrical parameters). In other cases, the geometrical accuracy will be the main requirement and productivity will be a secondary objective. In some other cases, only one geometrical parameter, for example the depth of the DES, will be critical and the other geometrical parameters should be kept only under certain thresholds, again by fixing a maximum relative error for these geometrical parameters. Therefore, this work considers the geometrical dimensions and the MRR that is actually obtained as its output, their deviance from the programmed values and the relative errors from the programmed to the real values. Table 5 summarizes all the calculated outputs, their units, ranges and the relationship they have with other input or output variables. In summary, the 162 different laser conditions that were tested provided 14 data sets of 162 instances each with 9 attributes and one output variable to be predicted.

3 Data Mining Techniques

In our study we consider the analysis of each output variable separately, by defining a one-dimensional regression problem for each case. In this type of formulation, we have an output variable, y, which is modeled as a function,

Table 5. Output Variables

	Variable	Units	Range	Relationship
$\mathbf{y_1}$	*Measured Volume*	$10^3 \times \mu m^3$	130 – 1701	Independent
$\mathbf{y_2}$	*Measured Depth*	μm	25 – 230.60	Independent
$\mathbf{y_3}$	*Measured Diameter*	μm	118.50 – 208.80	Independent
$\mathbf{y_4}$	*Measured Length*	μm	0 – 70.20	Independent
$\mathbf{y_5}$	*Measured MRR*	$10^3 \times \mu m^3/s$	12 – 121	y_1/x_8
$\mathbf{y_6}$	*Volume error*	$10^3 \times \mu m^3$	-980 – 596	$x_4 - y_1$
$\mathbf{y_7}$	*Depth error*	μm	-180.60 – 56.90	$x_1 - x_2$
$\mathbf{y_8}$	*Width error*	μm	-62.80 – -16.63	$2x_2 - y_3$
$\mathbf{y_9}$	*Length error*	μm	-25.50 – 208.80	$x_3 - y_4$
$\mathbf{y_{10}}$	*MRR error*	$10^3 \times \mu m^3/s$	-61 – 66	$x_9 - y_5$
$\mathbf{y_{11}}$	*Volume relative error*	dimensionless	-1.36 – 0.82	y_6/x_4
$\mathbf{y_{12}}$	*Depth relative error*	dimensionless	-3.61 – 0.63	y_7/x_1
$\mathbf{y_{13}}$	*Width relative error*	dimensionless	-0.55 – 0.10	$y_8/(2x_2)$
$\mathbf{y_{13}}$	*Length relative error*	dimensionless	-0.71 – 0.15	y_9/x_3

f, of a set of independent variables, x, called *attributes* in the conventional notation of data mining, where m is the number of *attributes*. The function is expressed as follows in Equation 1:

$$y_{estimated} = f(x),\ x \in R^m, y \in R \tag{1}$$

The aim of this work is to determine the most suitable regressor for this industrial problem. The selection is performed by comparing the Root Mean Squared Error (RMSE) of several regressors over the data set. Having a data collection of n pairs of real values $\{x_i, y_i\}_{i=1}^{i=n}$, the RMSE is an estimation of the expected difference between the real and the forecasted output by a regressor. It is expressed as the square root of the mean of the squares of the deviations, as shown in Equation 2.

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} (y_t - f(x_t))^2}{n}} \tag{2}$$

We tested a wide range of the main families of state-of-the-art regression techniques:

- Regressors based on functions. We use two of the most popular algorithms, Support Vector Regression (SVR) [35] and ANNs [36], and also Linear Regression [37], which has an easier formulation that allows direct physical interpretation of the models. We should note the widespread use of SVR [38], while ANNs have been successfully applied to a great variety of industrial modeling problems [39, 40, 41].
- Methods based on instance, specifically its most representative regressor, k-Nearest Neighbors Regressor [42]. In this kind of algorithm, it is not necessary to express an analytic relationship between the input variables

and the output that is modeled, an aspect that makes this approach totally different from the other. Instead of using an explicit formulation to obtain a prediction, it is calculated from a set values stored in the training phase.

- Decision trees based regressors. We have included these kinds of methods because they are used in the ensembles regressors, as explained in subsection 4.1.
- Ensemble techniques [43] of the most extended use in the literature. These kinds of regressors have been successfully applied to a wide variety of industrial problems [44, 45, 46, 47, 32].

3.1 Linear Regression

One of the most natural and simplest ways of expressing relations between a set of inputs and an output is by using a linear function. In this type of regressor the variable to forecast is given by a linear combination of the attributes, with predetermined weights [37], as detailed in Equation 3.

$$y^{(i)}_{estimated} = \sum_{j=0}^{k} (w_j \times x^{(i)}_j) \tag{3}$$

$y^{(i)}_{estimated}$ denotes the output of the i-th training instance, and $x^{(i)}_j$ the j-th attribute of the i-th instance. To calculate the most adequate weights w_j, the sum of the squares of the differences between real and forecasted output is minimized, following the expression given in Equation 4.

$$\sum_{i=0}^{n} [y_{(i)} - \sum_{j=0}^{k} (w_j \times x^{(i)}_j)] \tag{4}$$

We used an improvement to this original formulation, by selecting the attributes with the Akaike Information Criterion (AIC) [48] In Equation 5, we can see how the AIC is defined, where k is the number of free parameters of the model (i.e., the number of input variables considered) and P is the probability of the estimation fitting the training data. The aim is to obtain models that fit the training data, but with as few parameters as possible.

$$AIC = -2ln(P) + 2k \tag{5}$$

3.2 Decision trees based regressors

The decision tree is a data-mining technique that builds hierarchical models that are easily interpretable, because they may be represented in graphical form, as shown in Figures 2 and 3, with an example of the output *measured length* as a function of the input attributes. In this type of model, all the decisions are organised around a single variable, resulting in the final hierarchical

structure. This representation has three elements: the *nodes*, attributes taken for the decision (ellipses in the representation), the *leaves*, final forecasted values (squares), these two elements being connected by *arcs*, with the splitting values for each attribute.

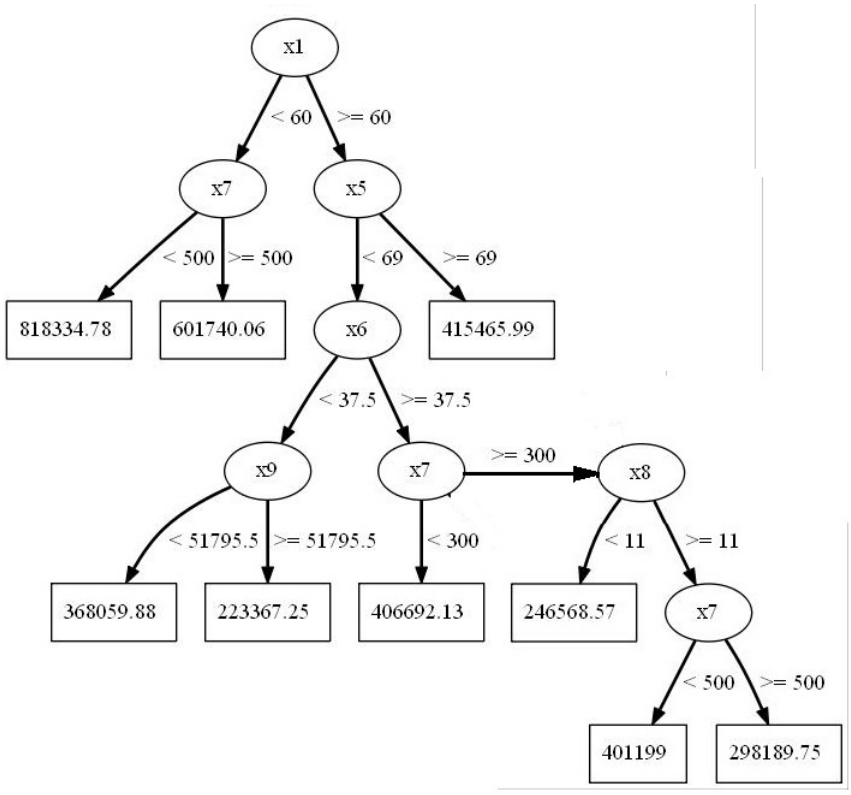


Fig. 2. Model of the Measured Volume with a Regression Tree

As base regressors, we have two types of regressors the structure of which is based on decision trees: *regression trees* and *model trees*. Both families of algorithms are hierarchical models represented by an abstract tree, but they differ with regard to what their leaves store [49]. In the case of *regression trees*, a value is stored that represents the average value of the instances enclosed by the leaf, while the *model trees* have a linear regression model that predicts the output value for these instances. The intrasubset variation in the class values down each branch are minimized to build the initial tree [50].

In our experimentation, we have used one representative implementation of the two families, Reduced-Error Pruning Tree (REPTree) [51], a regression

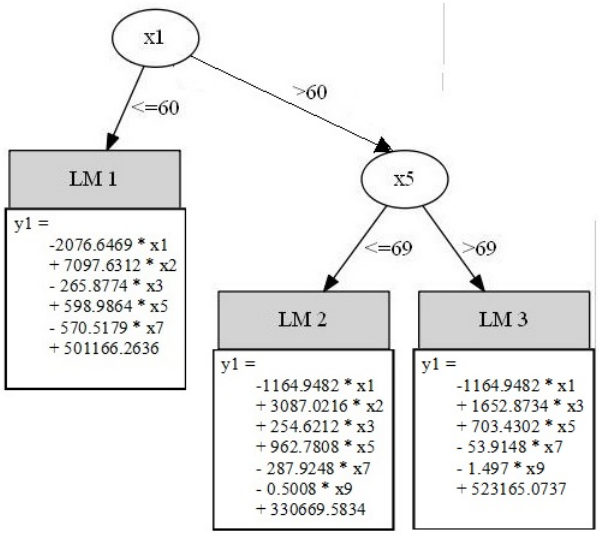


Fig. 3. Model of the Measured Volume with a Model Tree

tree, and M5P [49], a model tree. In both cases we have tested two configurations, pruned and unpruned trees. In the case of having one single tree as a regressor for some typologies of ensembles, it is more appropriate to prune the trees to avoid overfitting the training data, while some ensembles can take advantage of having unpruned trees [52].

3.3 Ensemble Regressors

An ensemble regressor combines the predictions of a set of so-called base regressors using a voting system [53], as we can see in Figure 4. Probably the three most popular ensemble techniques are Bagging [54], Boosting [55] and Random Subspaces [52]. For Bagging and Boosting, the ensemble regressor is formed from a set of weak base regressors, trained by applying the same learning algorithm to different sets obtained from the training set. In Bagging, each base regressor is trained with a dataset obtained from random sampling *with replacement* [54] (i.e., a particular instance may appear repeated several times or may not be considered in any base regressor). As a result, the base regressors are independent. However, Boosting uses all the instances and a set of weights to train each base regressor. Each instance has a weight pointing out how important it is to predict that instance correctly. Some base regressors can take weights into account (e.g., decision trees). Boosting trains base regressors sequentially, because errors for training instances in the previous base regressor are used for reweighting. The new base regressors are focused on instances that previous base regressors have wrongly predicted. The voting

system for Boosting is also weighted by the accuracy of each base regressor [56].

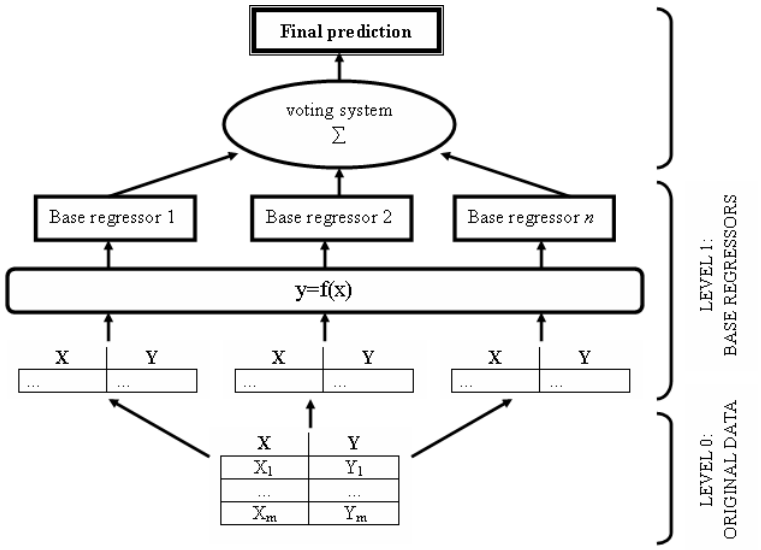


Fig. 4. Ensemble regressor architecture

Random Subspaces follow a different approach: each base regressor is trained in a subset of fewer dimensions than the original space. This subset of features is randomly chosen for all regressors. This procedure is followed with the intention of avoiding the well-known problem of the *curse of dimensionality*, suffered by many regressors when there are many features, and to improve accuracy by choosing base regressors with low correlations between them. In total, we have used five ensemble regression techniques of the state of the art for regression, two variants of Bagging, one variant of Boosting, and Random Subspaces. The list of ensemble methods used in the experimentation is enumerated below.

- Bagging in its initial formulation for regression
- IteratedBagging, which combines several Bagging ensembles, the first one keeping to a typical construction and the others using residuals (differences between the real and the predicted values) for training purposes [57].
- Random Subspaces, in its formulation for regression
- AdaBoost.R2 [58], a boosting implementation for regression. Calculated from the absolute errors of each training example, $l(i) = |f_R(x_i) - y_i|$, the so-called *loss function*, $L(i)$, was used, to estimate the error of each base regressor and to assign a suitable weight to each one. Let Den be the maximum value of $l(i)$ in the training set, then three different loss

functions are used: linear, $L_l(i) = l(i)/Den$, square, $L_S(i) = [l(i)/Den]^2$, and exponential, $L_E(i) = 1 - exp(-l(i)/Den)$.

- Additive Regression. This regressor has a learning algorithm called Stochastic Gradient Boosting [59], which is a modification of Adaptive Bagging, a hybrid Bagging Boosting procedure intended for least squares fitting on additive expansions [59].

3.4 *k*-Nearest Neighbor Regressor

This regressor is the most representative algorithm among the instance-based learning. These kinds of methods forecast the output value using stored values of the most similar instances of the training data [60]. The estimation is the mean of the k most similar training instances. Two configuration decisions have to be taken:

- How many nearest neighbors to use to forecast the value of a new instance.
- Which distance function is used to measure the similarity between the instances.

In our experimentation we have used the most common definition of the distance function, Euclidean distance, while the number of neighbors is optimized using cross validation.

3.5 Support Vector Regressor

This kind of regressor is based on a parametric function, whose parameters are optimized during the training process, in order to minimize the RMSE [61]. Mathematically, the goal is to find a function, $f(x)$, that has the most deviation, ϵ, from the targets that are actually obtained, y_i, for all the training data, and at the same time is as flat as possible. Equation 6 is an example of SVR with the particular case of a linear function, called linear SVR, where $< ., . >$ denotes the inner product in the input space, X.

$$f(x) =< w, x > +b \text{ with } w \in X \; b \in R \tag{6}$$

The norms of w have to be minimized to find a flat function, but in real data solving this optimization problem can be infeasible. In consequence, Boser et al [35] introduced three terms into the formulation: the slack variables ξ, ξ^* and C, a trade-off parameter between the flatness and the deviations of the errors larger than ϵ. In Equation 7, the optimization problem is shown that is associated with a linear SVR.

$$\begin{array}{l} \text{minimize } 1/2||w||^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*) \\ \text{s. t.} \\ y_i - < w, x_i > -b \leq \epsilon + \xi_i \\ < w, x_i > +b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \, \xi_i^* \geq 0 \end{array} \tag{7}$$

We have an optimization problem of the convex type that is solved in practice using the Lagrange method. The equations are re-written using the *primal* objective function and the corresponding constraints, a process in which the so-called *dual* problem (see Equation 8) is obtained.

$$\begin{aligned} &\text{maximize} \\ &-1/2 \sum_{i,j=1}^{l} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) < x_i, x_j > \\ &-\epsilon \sum_{i=1}^{l} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \\ &s.t \\ &\sum_{i=1}^{l} (\alpha_i - \alpha_i^*) = 0 \\ &\alpha_i,\ \alpha_i^* \in [0, C] \end{aligned} \tag{8}$$

From the expression in Equation 8, it is possible to generalize the formulation of the SVR in terms of a non-linear functions. Instead of calculating the inner products in the original feature space, a kernel function, $k(x, x')$, that computes the inner product in a transformed space was defined. This kernel function has to satisfy the so-called Mercer's conditions [62]. In our experimentation, we have used the two most popular kernels in the literature [38]: linear and radial basis.

3.6 Artificial Neural Networks

The Multilayer Perceptron (MLP), the most popular variation of ANNs [63], was used in the experimental work. It has been demonstrated to be a universal approximator of functions [64]. ANNs are a particular case of Neural Networks, the mathematical formulation of which is inspired by biological functions, as it aims to emulate the behavior of a set of neurons [63]. This network has three layers [65], one with the network inputs (features of the data set), a hidden layer, and an output layer where the prediction is assigned to each input instance. Firstly the output of the hidden layer, y_{hide}, is calculated from the inputs, and then the output, y_{hide}, is obtained according to the expressions shown in the Equation 9 [65]:

$$\begin{aligned} y_{hide} &= f_{net}(W_1 x + B_1) \\ y_{output} &= f_{output}(W_2 y_{hide} + B_2) \end{aligned} \tag{9}$$

where, W_1 is the weight matrix of the hidden layer, B_1 is the bias of hidden layer, W_2 the weight matrix of the output layer (e.g., the identity matrix in the configurations tested), B_2 is the bias of the output layer, f_{net} is the activation function of the hidden layer, and f_{output} is the activation function of the output layer. These two functions depend on the chosen structure, but are typically the identity for the hidden layer and *tansig* for the output layer [65].

4 Results and Discussion

We compared the RMSE obtained for the regressors in a 10×10 cross-validation, in order to chose the method that is most suited to model this industrial problem. The experiments were completed using the WEKA [51] implementation of the methods described above.

All the ensembles under consideration have 100 base regressors. The methods that depend on a set of parameters are optimized as follows.

- SVR with linear Kernel: The trade-off parameter, C, in the range 2–8
- SVR with radial basis Kernel: C from 1 to 16, and the parameter of the radial basis function, *gamma*, from 10^{-5} to 10^{-2}
- Multi-Layer Perceptron: The training parameters *momentum, learning rate and number of neurons* are optimized in the ranges 0.1–0.4, 0.1–0.6, 5–15
- kNN: the number of neighbours is optimized from 1 to 10

The notation used to describe the methods is detailed in Table 6.

Table 6. Methods Notation

Bagging	BG
Iterated Bagging	IB
Random Subspaces	RS
AdaboostR2	R2
Additive Regression	AR
REPTree	RP
M5P Model Tree	M5P
Support Vector Regressor	SVR
Multi Layer Perceptron	MLP
k-Nearest Neighbor Regressor	kNN
Linear Regression	LR

Regarding the notation, two abbreviations have been used besides those that are indicated in Table 6. On the one hand, we used the suffixes "L", "S" and "E" for the Linear, Square and Exponential loss functions of Adaboost.R2, and, on the other, the trees that are either pruned (P) or unpruned (U) appear between brackets. Tables 7 to 9 set out the RMSE of each of the 14 outputs for each method.

Finally, a summary table with the best methods per output is shown. The indexes of the 39 methods tested are explained in Tables 10 and 11, and in Table 12 the method with minimal RMSE is indicated, according to the notation for these indexes. In the third column, we also indicate those methods that have larger RMSE, but using a corrected resampled t-test [66], the differences are not statistically significative at a confidence level of 95%. Analyzing the

Table 7. Root Mean Squared Error. 1/3

	Volume	**Depth**	**Width**	**Length**	**MRR**
RP	216482.27	26.89	5.81	5.79	17166.73
M5P	210773.29	23.01	4.51	5.53	16076.33
LR	197521.89	23.84	7.04	14.73	15015.93
kNN	193756.62	23.57	5.42	6.84	15842.42
SVR Linear	197894.5	24.19	7.07	16.23	14800.32
SVR Radial Basis	200774.24	18.85	4.38	5.35	14603.37
MLP	207646.99	22.98	4.7	5.39	16292.43
BG RP P	200212.92	21.49	4.97	5.08	15526.79
BG RP U	205290.8	19.98	4.66	4.9	15603.89
BG M5P P	200636.57	20.61	4.43	5.27	15293.92
BG M5P U	197546.75	19.5	4.31	5.38	15022.1
IB RP P	202767.85	22.05	4.87	5.14	15677.57
IB RP U	219388.93	21.7	5.01	5.11	15911.96
IB M5P P	197833.83	20.8	4.42	4.88	15318.01
IB M5P U	195154.16	19.65	4.3	4.78	14830.83
R2-L RP P	191765.23	22.27	4.94	5.07	15788.47
R2-L RP U	206369.82	21.71	5.25	5.29	17004.31
R2-L M5P P	186843.92	20.81	4.37	4.84	15031.37
R2-L M5P U	181587.4	20.51	4.34	4.84	15209.01
R2-S RP P	193908.39	23.03	5.1	5.18	16007.43
R2-S RP U	200453.21	21.21	5.11	5.31	16401.24
R2-S M5P P	173117.72	20.83	4.49	4.71	15070.05
R2-S M5P U	173914.36	20.87	4.52	4.72	15245.81
R2-E RP P	192529.31	22.59	5.02	5.07	15854.03
R2-E RP U	205920.86	21.44	5.22	5.21	17318.45
R2-E M5P P	171056.22	21.25	4.53	4.66	15078.01
R2-E M5P U	172948.95	21.32	4.55	4.67	15090.23
AR RP P	215750.39	25.51	5.42	5.56	17249.82
AR RP U	274467.43	24.71	5.83	6.15	18628.28
AR M5P P	208805.84	22.27	4.44	5.08	16076.5
AR M5P U	194572.79	19.58	4.51	4.94	15538.29
RS 50% RP P	200526.36	26.58	5.68	7.74	15854.27
RS 50% RP U	201946.24	25.33	5.35	7.54	15669.28
RS 50% M5P P	201013.27	26.6	5.38	15.42	15402.77
RS 50% M5P U	199166.84	25.65	5.39	15.45	15349.84
RS 75% RP P	199270.03	23.33	5.27	5.98	15861.58
RS 75% RP U	207845.67	21.61	5.04	6.25	16251.64
RS 75% M5P P	199648.96	23.55	4.65	8.29	15227.87
RS 75% M5P U	197420.35	22.11	4.6	8.39	15295.71

Table 8. Root Mean Squared Error. 2/3

	Volume error	**Depth error**	**Width error**	**Length error**	**MRR error**
RP	216875.74	30.66	5.91	5.62	16355.57
M5P	214500.91	19.8	6.19	5.09	16009.84
LR	197521.93	23.92	6.76	14.67	14963.09
kNN	193696.36	23.69	4.98	6.6	14636.51
SVR Linear	197817.88	24.17	7.1	16.23	14784.77
SVR Radial Basis	200785.96	18.98	4.42	5.37	14504.61
MLP	206753.61	21.96	4.93	5.37	17382.69
BG RP P	201037.57	23.18	4.62	4.98	15122.57
BG RP U	205341.53	21.6	4.4	4.87	15327.2
BG M5P P	200594.31	19.66	5.32	5.08	15162.5
BG M5P U	197575.49	19.19	5.22	5.15	14860.54
IB RP P	207848	23.05	4.78	5.18	15525.56
IB RP U	216631.14	23.79	4.79	5.2	16532.31
IB M5P P	201450.65	19.73	4.44	4.71	15177.67
IB M5P U	198261.22	19.44	4.33	4.58	14933.58
R2-L RP P	201365.5	24.08	4.64	5.03	15442.66
R2-L RP U	208500.97	23.95	4.87	5.38	16778.3
R2-L M5P P	184799.5	20.61	4.68	4.77	14732.4
R2-L M5P U	183740.85	20.71	4.7	4.81	14817.65
R2-S RP P	195592.43	24.75	4.65	5.24	16107.62
R2-S RP U	201017.69	22.87	4.71	5.39	15871.67
R2-S M5P P	172775.15	21.09	4.53	4.72	14459.98
R2-S M5P U	173892.35	20.88	4.52	4.74	14497.38
R2-E RP P	195657.69	24.26	4.62	5.12	15645.02
R2-E RP U	206275.71	24.38	4.82	5.35	16721.26
R2-E M5P P	172196.89	21.61	4.57	4.75	14324.48
R2-E M5P U	173356.89	21.61	4.58	4.79	14371.36
AR RP P	214978.3	27.77	5.49	5.43	16208.07
AR RP U	271926.55	27.43	5.29	6.1	20571.27
AR M5P P	211329.63	19.8	4.55	4.66	15995.84
AR M5P U	195497.5	19.55	4.78	4.82	15448.38
RS 50% RP P	200775.17	28.54	5.41	6.93	15217.65
RS 50% RP U	202354.2	26.84	5.24	6.94	15074.78
RS 50% M5P P	201054.45	25.91	5.98	11.81	15387.34
RS 50% M5P U	199246.1	25.61	5.79	11.75	14940.89
RS 75% RP P	199501.89	25.86	4.9	5.73	15165.01
RS 75% RP U	206467.56	24.5	4.75	6.06	15743.5
RS 75% M5P P	200160.47	22.05	5.76	6.66	15185.2
RS 75% M5P U	197431.78	21.79	5.53	6.75	14729.75

Table 9. Root Mean Squared Error. 3/3

	Volume rel. error	**Depth rel. error**	**Width rel. error**	**Length rel. error**
RP	0.3	0.54	0.04	0.09
M5P	0.3	0.33	0.04	0.09
LR	0.27	0.43	0.05	0.13
kNN	0.27	0.38	0.04	0.1
SVR Linear	0.27	0.44	0.05	0.15
SVR Radial Basis	0.28	0.29	0.04	0.1
MLP	0.29	0.34	0.04	0.1
BG RP P	0.28	0.4	0.04	0.08
BG RP U	0.29	0.36	0.04	0.08
BG M5P P	0.28	0.32	0.04	0.09
BG M5P U	0.27	0.31	0.04	0.09
IB RP P	0.29	0.38	0.04	0.09
IB RP U	0.3	0.38	0.04	0.09
IB M5P P	0.28	0.32	0.04	0.09
IB M5P U	0.28	0.3	0.04	0.09
R2-L RP P	0.27	0.39	0.04	0.08
R2-L RP U	0.29	0.38	0.04	0.1
R2-L M5P P	0.25	0.32	0.04	0.09
R2-L M5P U	0.26	0.32	0.04	0.09
R2-S RP P	0.27	0.39	0.04	0.09
R2-S RP U	0.28	0.38	0.04	0.1
R2-S M5P P	0.24	0.32	0.04	0.1
R2-S M5P U	0.24	0.33	0.04	0.1
R2-E RP P	0.27	0.4	0.04	0.09
R2-E RP U	0.29	0.38	0.04	0.11
R2-E M5P P	0.24	0.33	0.04	0.1
R2-E M5P U	0.24	0.33	0.04	0.1
AR RP P	0.3	0.47	0.04	0.09
AR RP U	0.38	0.44	0.05	0.11
AR M5P P	0.29	0.33	0.04	0.09
AR M5P U	0.27	0.29	0.04	0.09
RS 50% RP P	0.28	0.48	0.04	0.09
RS 50% RP U	0.28	0.44	0.04	0.09
RS 50% M5P P	0.28	0.43	0.04	0.09
RS 50% M5P U	0.28	0.42	0.04	0.09
RS 75% RP P	0.28	0.45	0.04	0.08
RS 75% RP U	0.29	0.41	0.04	0.08
RS 75% M5P P	0.28	0.36	0.04	0.09
RS 75% M5P U	0.27	0.35	0.04	0.09

second column of Table 12, among the 39 configurations tested, only 4 methods obtained the best RMSE for one of the 14 outputs: Adaboost.R2 with exponential loss and pruned M5P as base regressors –index 26– (5 times), SVR with radial basis function kernel –index 6– (4 times), Iterated Bagging with unpruned M5P as base regressors –index 15– (4 times) and Bagging with unpruned RP as the base regressor –index 9– (1 time).

Table 10. Index Notation for the non-ensemble methods

1	2	3	4	5	6	7
RP	M5P	LR	kNN	SVR Linear	SVR Radial Basis	MLP

Table 11. Index Notation for the ensemble methods

	BG	IB	R2-L	R2-S	R2-E	AR	RS 50%	RS 75%
RP P	8	12	16	20	24	28	32	36
RP U	9	13	17	21	25	29	33	37
M5P P	10	14	18	22	26	30	34	38
M5P U	11	15	19	23	27	31	35	39

The performance of each method may be ranked. Table 13 presents the numbers of significative weaker performances of each method, considering the 14 outputs modeled. The most robust method is Iterated Bagging with unpruned M5P as base regressors –index 15–, as in none of the 14 outputs was it outperformed by other methods. Besides selecting the best method, it is possible to obtain some additional conclusions from this ranking table:

- Linear models like SVR Linear –index 5– and LR –index 3– do not fit the datasets in the study very well. Both methods are ranked together at the middle of the table. The predicted variables therefore need methods that can operate with nonlinearities.
- SVR with Radial Basis Function Kernel –index 6– is the only non-ensemble method with competitive results, but needs to tune 2 parameters. MLP –index 7– is not a good choice. It needs to tune 3 parameters and is not a well ranked method.
- For some ensemble configurations there are differences in the number of statistically significative performances between using pruned or unpruned trees, while in other cases these differences do not exist, but in general using unpruned trees is more accurate, specially in top-ranked methods. In fact, the only regressor which is not outperformed by other methods in any output has unpruned trees. Unpruned trees are more sensitive to changes in the training set. So, the predictions of unpruned trees, when their base

Table 12. Summary table

	Best method	**Statistically equivalent**
volume	**26**	3, 4, 5, **6**, 7, 11, 14, **15**, 16, 18, 19 20, 22, 23, 27, 31, 35, 38, 39
depth	**6**	9, 10, 11, 13, 14, **15**, 17, 18, 19, 21, 22 23, 25, 31
Width	**15**	2, **6**, 7, **9**, 10, 11, 12, 14, 18, 19, 22 23, **26**, 27, 30, 31, 39
Length	**26**	4, **6**, **9**, 12, 13, 14, **15**, 16, 17, 18, 19 22, 23, 24, 25, 27, 30, 31, 37
MRR	**6**	2, 3, 4, 5, 7, 8, **9**, 10, 11, 12, 13 14, **15**, 16, 17, 18, 19, 20, 21, 22, 23, 24 25, **26**, 27, 30, 31, 32, 33, 34, 35, 36, 38, 39
Volume error	**26**	3, 4, 5, **6**, 10, 11, **15**, 18, 19, 22, 23 27, 31, 34, 35, 36, 38, 39
Depth error	**6**	2, 10, 11, 14, **15**, 18, 19, 22, 23, **26**, 27 30, 31
Width error	**15**	6, 8, **9**, 12, 13, 14, 16, 17, 18, 19, 20 21, 22, 23, 24, 25, **26**, 27, 30, 37
Length	**15**	4, 8, **9**, 14, 16, 18, 19, 22, 23, **26**, 27 30, 31
MRR error	**26**	1, 2, 3, 4, 5, **6**, 8, **9**, 10, 11, 12 13, 14, **15**, 16, 17, 18, 19, 20, 21, 22, 23 24, 25, 27, 28, 30, 31, 32, 33, 34, 35, 36 37, 38, 39
Volume rel. error	**26**	3, 4, 5, **6**, 10, 11, **15**, 18, 19, 22, 23 27, 31, 34, 35, 36, 38, 39
Depth rel. error	**6**	2, 10, 11, 14, **15**, 18, 19, 22, 23, **26**, 27 30, 31
Width rel. error	**15**	2, **6**, 7, **9**, 10, 11, 14, 24, 30, 31, 38, 39
Length rel. error	**9**	1, 2, 8, 10, 11, 12, 13, 14, **15**, 16, 20 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39

regressors are trained in an ensemble, are more likely to output diverse predictions. If the predictions of all base regressors agreed there would be little benefit in using ensembles. Diversity balances faulty predictions by some base regressors with correct predictions by others.

- The top-ranked ensembles use the most accurate base regressor (i.e., M5P). All M5P configurations have a fewer weaker performances than the corresponding RP configuration. In particular, the lowest rank was assigned to the AR – RP configurations, while AR M5P U came second best.
- Ensembles that lose a lot information, such as RS, are ranked at the bottom of the table. The table shows that the lower the percentage of features RS use, the worse they perform. In comparison to other ensemble methods,

RS is a method that is very insensitive to noise, so it can point to data that are not noisy.

- In R2 M5P ensembles, the loss function does not appear to be an important configuration parameter
- IB M5P U is the only configuration that never had significant losses when compared with the other methods.

Table 13. Methods ranking

Indexes	**Methods**	**Number of defeats**
15	IB M5P U	0
31	AR M5P U	1
6, 14, 18, 19, 22, 23	SVR Radial Basis, IB M5P P, R2-L M5P P, R2-L M5P U, R2-S M5P P, R2-S M5P U	2
11, 26, 27	BG M5P U, R2-E M5P P, R2-E M5P U	3
10, 30	BG M5P P, AR M5P P	4
9	BG RP U	5
39	RS 75% M5P U	6
2, 4, 16, 38	M5P, kNN, R2-L RP P, RS 75% M5P P	7
12, 13, 35	IB RP P, IB RP U, RS 50% M5P U	8
3, 5, 8, 17, 20, 24, 25, 34, 36	LR, SVR Linear, BG RP P, R2-L RP U, R2-S RP P, R2-E RP P, R2-E RP U, RS 50% M5P P, RS 75 % RP P	9
7, 21, 37	MLP, R2-S RP U, RS 75% RP U	10
32, 33	RS 50% RP P, RS 50% RP U	11
1, 28	RP, AR RP P	12
29	AR RP U	14

5 Conclusions

In this study, extensive modeling has been presented with different data-mining techniques for the prediction of geometrical dimensions and productivity in the laser-milling of micro-cavities for the manufacture of drug eluting stents. Experiments on 316L Stainless Steel have been performed to provide data for the models. The experiments change most of the process parameters that can be changed under industrial conditions: scanning speed, laser pulse intensity and laser pulse frequency, moreover 2 different geometries and 3 different sizes were manufactured within the experimental test to obtain extensive data sets for this industrial task. Besides, a very extensive characterization and analysis of the results of the experimental test were performed to cover all the possible optimization strategies that industry might require

for DES manufacturing: from high-productivity objectives to high geometrical accuracy in just one geometrical axis. By doing so, 14 data sets were generated, each of 162 instances.

The experimental test clearly outlined that the geometry of the feature to be machined will affect the performance of the milling process, but also that is not easy to find the proper combination of process parameters to achieve the final part, which makes it clear that the laser micromilling of such geometries is a complex process to control. Therefore the use of data-mining techniques is proposed for the prediction and optimization of this process. Each variable to predict has been modelled, using regression methods, to forecast a continuous variable.

The paper shows an exhaustive test covering 39 regression method configurations for the 14 output variables. A 10×10 cross validation was used in the test to reveal the methods with a relatively better RMSE. A corrected resampled t-test was used to estimate significative differences. The test showed that ensemble regression techniques using M5P unpruned trees gave a better performance than other well-established soft computing techniques such as ANNs and Linear Regression techniques. SVR with Radial Basis Function Kernel was also a very competitive method, but required parameter tuning. We propose the use of an Iterated Bagging technique with an M5P unpruned tree as a base regressor, because its RMSE was never significantly worse than the RMSE of any of the other methods for any of the 14 variables.

Future work will consider applying the experimental procedure to different polymers, magnesium and other biodegradable and biocompatible elements. Also, as micromachining is a complex process where many variables play an important role in the geometrical dimensions of the machined workpiece, the application of visualization techniques, such as scatter plot matrices and start plots, to evaluate the relationships between inputs and outputs, will help us to understand this promising machining process more clearly, at an industrial level.

Acknowledgments

The authors would like to express their gratitude to the GREP research group at the University of Girona and the Tecnológico de Monterrey for the facilities provided during the experiments. This work was partially funded through grants from the IREBID project (FP7-PEOPLE-2009-IRSES-247476) of the European Commission and projects TIN2011-24046 and TECNIPLAD (DPI2009- 09852) pf tje the Spanish Science and Innovation Ministry .

References

1. P. A. Sugioka K., Meunier M., *Laser precision microfabrication.* Springer, 2010, vol. 135.

2. S. Garg and P. W. Serruys, "Coronary stentscurrent status," *Journal of the American College of Cardiology*, vol. 56, no. 10s1, pp. S1–S42, 2010.
3. D. M. Martin and F. J. Boyle, "Drug-eluting stents for coronary artery disease: a review," *Medical engineering & physics*, vol. 33, no. 2, pp. 148–163, 2011.
4. S. Campanelli, G. Casalino, and N. Contuzzi, "Multi-objective optimization of laser milling of 5754 aluminum alloy," *Optics & Laser Technology*, vol. 52, pp. 48–56, 2013.
5. J. Ciurana, G. Arias, and T. Ozel, "Neural network modeling and particle swarm optimization (pso) of process parameters in pulsed laser micromachining of hardened aisi h13 steel," *Materials and Manufacturing Processes*, vol. 24, no. 3, pp. 358–368, 2009.
6. J. Cheng, W. Perrie, S. Edwardson, E. Fearon, G. Dearden, and K. Watkins, "Effects of laser operating parameters on metals micromachining with ultrafast lasers," *Applied Surface Science*, vol. 256, no. 5, pp. 1514–1520, 2009.
7. I. E. Saklakoglu and S. Kasman, "Investigation of micro-milling process parameters for surface roughness and milling depth," *The International Journal of Advanced Manufacturing Technology*, vol. 54, no. 5-8, pp. 567–578, 2011.
8. D. Ashkenasi, T. Kaszemeikat, N. Mueller, R. Dietrich, H. J. Eichler, and G. Illing, "Laser trepanning for industrial applications," *Physics Procedia*, vol. 12, pp. 323–331, 2011.
9. B. Yilbas, S. Akhtar, and C. Karatas, "Laser trepanning of a small diameter hole in titanium alloy: Temperature and stress fields," *Journal of Materials Processing Technology*, vol. 211, no. 7, pp. 1296–1304, 2011.
10. R. Biswas, A. Kuar, S. Sarkar, and S. Mitra, "A parametric study of pulsed nd: Yag laser micro-drilling of gamma-titanium aluminide," *Optics & Laser Technology*, vol. 42, no. 1, pp. 23–31, 2010.
11. L. Tricarico, D. Sorgente, and L. D. Scintilla, "Experimental investigation on fiber laser cutting of ti6al4v thin sheet," *Advanced Materials Research*, vol. 264, pp. 1281–1286, 2011.
12. N. Muhammad, D. Whitehead, A. Boor, W. Oppenlander, Z. Liu, and L. Li, "Picosecond laser micromachining of nitinol and platinum–iridium alloy for coronary stent applications," *Applied Physics A*, vol. 106, no. 3, pp. 607–617, 2012.
13. H. Meng, J. Liao, Y. Zhou, and Q. Zhang, "Laser micro-processing of cardiovascular stent with fiber laser cutting system," *Optics & Laser Technology*, vol. 41, no. 3, pp. 300–302, 2009.
14. T.-C. Chen and R. B. Darling, "Laser micromachining of the materials using in microfluidics by high precision pulsed near and mid-ultraviolet nd: Yag lasers," *Journal of materials processing technology*, vol. 198, no. 1, pp. 248–253, 2008.
15. D. Bruneel, G. Matras, R. Le Harzic, N. Huot, K. König, and E. Audouard, "Micromachining of metals with ultra-short ti-sapphire lasers: Prediction and optimization of the processing time," *Optics and Lasers in Engineering*, vol. 48, no. 3, pp. 268–271, 2010.
16. M. Pfeiffer, A. Engel, S. Weißmantel, S. Scholze, and G. Reisse, "Microstructuring of steel and hard metal using femtosecond laser pulses," *Physics Procedia*, vol. 12, pp. 60–66, 2011.
17. D. Karnakis, M. Knowles, P. Petkov, T. Dobrev, and S. Dimov, "Surface integrity optimisation in ps-laser milling of advanced engineering materials," *Lasers in Manufacturing LIM*, 2007.
18. H. Qi and H. Lai, "Micromachining of metals and thermal barrier coatings using a 532nm nanosecond fiber laser," *Physics Procedia*, vol. 39, pp. 603–612, 2012.

19. D. Teixidor, I. Ferrer, J. Ciurana, and T. Özel, "Optimization of process parameters for pulsed laser milling of micro-channels on aisi h13 tool steel," *Robotics and Computer-Integrated Manufacturing*, 2012.
20. A. J. Torabi, E. M. Joo, L. Xiang, L. B. Siong, Z. Lianyin, P. S. Jie, Z. Junhong, S. Lin, H. Sheng, and J. T. T. Tijo, "A survey on artificial intelligence technologies in modeling of high speed end-milling processes," in *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on.* IEEE, 2009, pp. 320–325.
21. M. Chandrasekaran, M. Muralidhar, C. M. Krishna, and U. Dixit, "Application of soft computing techniques in machining performance prediction and optimization: a literature review," *The International Journal of Advanced Manufacturing Technology*, vol. 46, no. 5-8, pp. 445–464, 2010.
22. A. K. Choudhary, J. A. Harding, and M. K. Tiwari, "Data mining in manufacturing: a review based on the kind of knowledge," *Journal of Intelligent Manufacturing*, vol. 20, no. 5, pp. 501–521, 2009.
23. A. K. Dubey and V. Yadava, "Laser beam machining – a review," *International Journal of Machine Tools and Manufacture*, vol. 48, no. 6, pp. 609–628, 2008.
24. B. F. Yousef, G. K. Knopf, E. V. Bordatchev, and S. K. Nikumb, "Neural network modeling and analysis of the material removal process during laser machining," *The International Journal of Advanced Manufacturing Technology*, vol. 22, no. 1-2, pp. 41–53, 2003.
25. S. Campanelli, G. Casalino, A. Ludovico, and C. Bonserio, "An artificial neural network approach for the control of the laser milling process," *The International Journal of Advanced Manufacturing Technology*, pp. 1–8, 2012.
26. C. Jimin, Y. Jianhua, Z. Shuai, Z. Tiechuan, and G. Dixin, "Parameter optimization of non-vertical laser cutting," *The International Journal of Advanced Manufacturing Technology*, vol. 33, no. 5-6, pp. 469–473, 2007.
27. D. Teixidor, M. Grzenda, A. Bustillo, and J. Ciurana, "Modeling pulsed laser micromachining of micro geometries using machine-learning techniques," *Journal of Intelligent Manufacturing*, pp. 1–14, 2013.
28. N. C. Oza and K. Tumer, "Classifier ensembles: Select real-world applications," *Information Fusion*, vol. 9, no. 1, pp. 4–20, 2008.
29. P. Santos, L. Villa, A. Reñones, A. Bustillo, and J. Maudes, "Wind turbines fault diagnosis using ensemble classifiers," *Advances in Data Mining. Applications and Theoretical Aspects*, vol. 7377, pp. 67–76, 2012.
30. A. Bustillo and J. J. Rodríguez, "Online breakage detection of multitooth tools using classifier ensembles for imbalanced data," *International Journal of Systems Science*, no. ahead-of-print, pp. 1–13, 2013, doi: 10.1080/00207721.2013.775378.
31. J.-F. Díez-Pastor, A. Bustillo, G. Quintana, and C. García-Osorio, "Boosting projections to improve surface roughness prediction in high-torque milling operations," *Soft Computing*, vol. 16, no. 8, pp. 1427–1437, 2012.
32. A. Bustillo, E. Ukar, J. J. Rodriguez, and A. Lamikiz, "Modelling of process parameters in laser polishing of steel components using ensembles of regression trees," *International Journal of Computer Integrated Manufacturing*, vol. 24, no. 8, pp. 735–747, 2011.
33. A. Bustillo, J.-F. Díez-Pastor, G. Quintana, and C. García-Osorio, "Avoiding neural network fine tuning by using ensemble learning: application to ball-end milling operations," *The International Journal of Advanced Manufacturing Technology*, vol. 57, no. 5, pp. 521–532, 2011.

34. S. Ferreiro, B. Sierra, I. Irigoien, and E. Gorritxategi, "Data mining for quality control: Burr detection in the drilling process," *Computers & Industrial Engineering*, vol. 60, no. 4, pp. 801–810, 2011.
35. B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.
36. R. Beale and T. Jackson, *Neural computing: an introduction*. Bristol, UK, UK: IOP Publishing Ltd., 1990. [Online]. Available: http://portal.acm.org/citation.cfm?id=121342
37. A. Sykes, *An introduction to regression analysis*. Law School, University of Chicago, 1993.
38. X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip *et al.*, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.
39. N. Tosun and L. Özler, "A study of tool life in hot machining using artificial neural networks and regression analysis method," *Journal of Materials Processing Technology*, vol. 124, no. 1, pp. 99–104, 2002.
40. A. Azadeh, S. Ghaderi, and S. Sohrabkhani, "Annual electricity consumption forecasting by neural network in high energy consuming industrial sectors," *Energy Conversion and Management*, vol. 49, no. 8, pp. 2272–2278, 2008.
41. P. Palanisamy, I. Rajendran, and S. Shanmugasundaram, "Prediction of tool wear using regression and ann models in end-milling operation," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 1-2, pp. 29–41, 2008.
42. S. Vijayakumar and S. Schaal, "Approximate nearest neighbor regression in very high dimensions," *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice, MIT Press, Cambridge, MA, USA*, pp. 103–142, 2006.
43. L. Kuncheva, "Combining classifiers: Soft computing solutions," *Pattern Recognition: From Classical to Modern Approaches*, pp. 427–451, 2001.
44. J. Yu, L. Xi, and X. Zhou, "Identifying source (s) of out-of-control signals in multivariate manufacturing processes using selective neural network ensemble," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 1, pp. 141–152, 2009.
45. T. W. Liao, F. Tang, J. Qu, and P. Blau, "Grinding wheel condition monitoring with boosted minimum distance classifiers," *Mechanical Systems and Signal Processing*, vol. 22, no. 1, pp. 217–232, 2008.
46. S. Cho, S. Binsaeid, and S. Asfour, "Design of multisensor fusion-based tool condition monitoring system in end milling," *The International Journal of Advanced Manufacturing Technology*, vol. 46, no. 5-8, pp. 681–694, 2010.
47. S. Binsaeid, S. Asfour, S. Cho, and A. Onar, "Machine ensemble approach for simultaneous detection of transient and gradual abnormalities in end milling using multisensor fusion," *Journal of Materials Processing Technology*, vol. 209, no. 10, pp. 4728–4738, 2009.
48. H. Akaike, "A new look at the statistical model identification," *Automatic Control, IEEE Transactions on*, vol. 19, no. 6, pp. 716–723, 1974.
49. J. R. Quinlan, "Learning with continuous classes," in *Proceedings of the 5th Australian joint Conference on Artificial Intelligence*, vol. 92. Singapore, 1992, pp. 343–348.

50. T. M. Khoshgoftaar, E. B. Allen, and J. Deng, "Using regression trees to classify fault-prone software modules," *Reliability, IEEE Transactions on*, vol. 51, no. 4, pp. 455–462, 2002.
51. I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, 2005, url: http://www.cs.waikato.ac.nz/ml/weka.
52. T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.
53. T. G. Dietterichl, "Ensemble learning," *The handbook of brain theory and neural networks*, pp. 405–408, 2002.
54. L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
55. Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *ICML*, vol. 96, 1996, pp. 148–156.
56. Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.
57. L. Breiman, "Using iterated bagging to debias regressions," *Machine Learning*, vol. 45, no. 3, pp. 261–277, 2001.
58. H. Drucker, "Improving regressors using boosting techniques," in *ICML*, vol. 97, 1997, pp. 107–115.
59. J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.
60. D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine learning*, vol. 6, no. 1, pp. 37–66, 1991.
61. A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.
62. C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
63. J. E. Dayhoff and J. M. DeLeo, "Artificial neural networks," *Cancer*, vol. 91, no. S8, pp. 1615–1635, 2001.
64. K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.
65. W. H. Delashmit and M. T. Manry, "Recent developments in multilayer perceptron neural networks," in *Proceedings of the seventh Annual Memphis Area Engineering and Science Conference, MAESC*, 2005.
66. C. Nadeau and Y. Bengio, "Inference for the generalization error," *Machine Learning*, vol. 52, no. 3, pp. 239–281, 2003.

5.2. Mantenimiento de aerogeneradores

A continuación se exponen las tres publicaciones del presente trabajo referentes al mantenimiento de aerogeneradores: "*Wind turbines fault diagnosis using ensemble classifiers*", "*An SVM-Based Solution for Fault Detection in Wind Turbines*" y "*Identifying maximum imbalance in datasets for fault diagnosis of gearboxes*".

Se han utilizado datos de un banco de pruebas del área de Diagnóstico Industrial y Mantenimiento Predictivo de la Fundación CARTIF para simular el conjunto tren de potencia del aerogenerador+viento. Sobre el banco de ensayos se registran señales de vibración a través de cuatro acelerómetros. Los fallos que se simularon en la bancada de ensayos fueron cuatro niveles de desequilibro (DE1, DE2, DE3 y DE4) y tres niveles de desalineamiento (DA1 y DA2). En total se tienen 15 situaciones posibles (una de funcionamiento correcto y 14 posibles fallos), multiplicando cinco posibles valores del desequilibrio (sin fallo y los cuatro niveles de fallo) por tres posibles valores de desalineamiento (sin fallo y los dos niveles de fallo). Sin embargo, en la práctica, la relación existente entre el desequilibrio y el desalineamiento hace que sólo se considerasen siete combinaciones de fallos en la experimentación (las indicadas en el Cuadro 5.7). El problema tiene por tanto ocho clases, los siete tipos de fallos y el funcionamiento correcto del aerogenerador.

En total se dispone de 6551 instancias para modelar el error de la turbina. En cada una de esas instancias se registran 544 atributos en función de los que se va a construir un clasificador para diagnosticar el funcionamiento del aerogenerador. Dichos atributos son el tiempo, las variables dinámicas, que describen el régimen de funcionamiento de la turbina (par y velocidad), cuatro medidas de la intensidad, que se mide tanto en el rotor como en el estator, y variables referidas al análisis de vibraciones de la turbina. Respecto a estos últimas variables, se tienen tres grupos: ordenes del sistema, parámetros estadísticos de la vibración y frecuencias naturales del sistema. Las variables estadísticas que se han tomado para resumir la distribución son las siguientes: media, RMS, skewness, kurtosis y rango intercuartil. En el Cuadro 5.5 se recogen los atributos del problema estudiado.

La primera de las publicaciones es un estudio preliminar de los datos en el que se analizó la metodología más adecuada para obtener una representación del problema abordable por técnicas de Minería de Datos. Una vez obtenida dicha representación, la segunda publicación presenta un análisis exhaustivo de la precisión que pueden obtener aproximadores universales de funciones (redes neuronales y tres tipos de SVM, con kernel gaussiano

Cuadro 5.5: Atributos considerados para el diagnóstico de fallos de aerogeneradores

Variable	**Unidades**	**Rango**
Par	% de par máximo	0 - 100
Velocidad	Hz	0,283 - 0,512
Corriente de entrada	Amperios	1,630 - 2,850
Corriente eléctrica en los ejes	Amperios	0 - 0,124
Armónicos	mm/s^2	0 - 0,075
Bandas	mm/s^2	0 - 0,052
Media	mm/s^2	0 - 0,004
RMS	mm/s^2	0,0016 - 0,121
Skewness	adimensional	1,400 - 4,780
Kurtosis	adimensional	2,190 - 66,100
Rango intercuartil	mm/s^2	0,021 - 0,168

y dos kernels con infinitas hipótesis) con los datos obtenidos. El Cuadro 5.6 muestra la comparativa del porcentaje de acierto que obtienen las distintas técnicas de clasificación en una validación cruzada 5×2. También se detallan los tiempos de entrenamiento y de ajuste de parámetros para los clasificadores. Teniendo en cuenta los tres parámetros (acierto y tiempos de entrenamiento y ajuste de parámetros), el mejor método es un SVM con kernel lineal. En todos los casos se indica con un asterisco aquellos casos que son peores de forma estadísticamente significativa que dicho método de acuerdo a un test *t* remuestreado corregido. Si bien la precisión de las redes neuronales y de un SVM con kernel stump no fue peor de forma estadísticamente significativa que la de un SVM con kernel lineal, sí lo fueron los tiempos de entrenamiento y ajuste. En la publicación "*An SVM-Based Solution for Fault Detection in Wind Turbines*" también se analiza la causa de que un SVM con kernel lineal pueda igualar a clasificadores mucho más complejos, llevando a la conclusión de que la definición de los atributos tiende a definir conjuntos linealmente separables.

Además del porcentaje de acierto, se analizó si había información redundante aplicando dos tipos de técnicas de selección de variables, una selección de tipo *filtro* y una selección de tipo *envoltorio*. La precisión obtenida por un SVM lineal con todas las variables y con las variables seleccionadas por ambas técnicas es la mostrada en el Cuadro 5.8. Con ambos métodos de selección de variables se pierde información de forma estadísti-

Cuadro 5.6: Comparación de SVMs y Redes neuronales

	SVM lineal	**SVM perceptrón**	**SVM gaussiano**	**SVM stump**	**Redes neuronales**
Acierto (%)	**98,26**	96,86 *	97,25 *	**97,85**	**97,47**
Tiempo ajuste parámetros (s)	444,27	1241,27 *	16068,97 *	945,17 *	46611,12 *
Tiempo entrenamiento (s)	**12,48**	21,55 *	15,61 *	22,17 *	491,00 *

camente significativa, lo que indica que no hay redundancia en los datos y que todas las variables son necesarias para obtener el porcentaje de acierto más alto.

Finalmente, se muestra una descomposición detallada del error de predicción entre las distintas clases, mediante la matriz de confusión para un SVM lineal (Cuadro 5.7). Dicha matriz indica lo siguiente:

1. No hubo error al distinguir entre fallos de desequilibrio y fallos de desalineamiento, ni entre esos dos fallos aislados frente a la combinación de ambos.

2. Ocurrieron algunos errores con muy baja frecuencia (únicamente el 0,02 %), tales como identificar un desequilibrio de nivel 2 cuando en realidad no se produjo ningún fallo o no identificar ningún fallo cuando realmente sí lo hubo (desequilibrios de nivel 1 y 4).

3. Casi todos los errores en la matriz de confusión se concentran en la distinción entre niveles de desequilibrio. Específicamente, el 90 % de todos los errores aparece en la discriminación entre el nivel de desequilibro 3 y el 4 (0,48 % del total de casos). Este hecho se explica por los pesos mediante los cuales se simularon los fallos en la bancada de ensayos. Mientras que el peso aproximadamente se duplicó para pasar del desequilibrio 1 al 2 y del 2 a 3, esta relación de pesos se cambió para el desequilibrio 4, donde sólo se subió el peso un 50 % respecto al desequilibrio 3.

Por otro lado, la tercera publicación se centra en la aplicación real del sistema de diagnóstico teniendo en cuenta el problema del desequilibrio en cuanto el número de instancias de cada clase. En primer lugar se compararon las distintas métricas para problemas de clasificación desequilibrados multiclase, concluyendo que la más adecuada desde un punto

Cuadro 5.7: Matriz de confusión para un SVM lineal

Predicho / **Real**	*NO*	*DE1*	*DE2*	*DE3*	*DE4*	*DA1*	*DA2*	*DE4+DA2*
NO	13,52	0	0,02	0	0	0	0	0
DE1	0,01	12,73	0	0,16	0,03	0	0	0
DE2	0	0	13,02	0,04	0	0	0	0
DE3	0	0,14	0,04	11,85	0,76	0	0	0
DE4	0,01	0,04	0	0,48	12,66	0	0	0
DA1	0	0	0	0	0	13,31	0	0
DA2	0	0	0	0	0	0	12,75	0
DE4+DA2	0	0	0	0	0	0	0	8,43

Cuadro 5.8: Precisión de un SVM lineal con selección de variables

Selección tipo filtro	**Selección tipo envoltorio**	**SVM con todas las variables**
95,57 *	95,17 *	**98,02**

de vista industrial para el diagnóstico de fallos en aerogeneradores en la macro media del índice F (MacroF). Posteriormente, se comparó mediante validaciones cruzadas 5×2 la precisión de distintas técnicas de clasificación en términos de la métrica MacroF, tanto de las diversas familias específicas de problemas de clasificación desequilibrados presentes en el estado del arte, como de técnicas generales de clasificación. Debido al gran número de clasificadores, se dividió la experimentación en 8 grupos de técnicas:

- Undersampling (UND).
- Oversampling (OVE).
- Ensemble de binarización de clasificadores binarios NO orientados a problemas desequilibrados (BIN-NO).
- Ensemble de binarización de clasificadores binarios orientados a problemas desequilibrados (BIN-DES).
- Ensemble de binarización + Undersampling (BIN+UND).
- Ensemble de binarización + Oversampling (BIN+OVE).

Cuadro 5.9: Configuración de cada experimento

UND	**OVE**	**BIN-NO**	**BIN-DES**	**BIN+OVE**	**BIN+UND**	**COS**	**GEN**
General	General	Binarios	Binarios-desequilibrado	Binarios	Binarios	General	General

- Clasificadores sensibles al coste (COS).
- Clasificadores de uso general, sin ninguna técnica específica de problemas multiclase desequilibrados (GEN).

Para cada grupo de técnicas, los clasificadores considerados son los que se indican en el Cuadro 5.9, y que se corresponden con lo siguiente:

- El grupo de clasificadores *binario* está formado por árboles de decisión C4.5 y C4.4 y por un SVM de kernel lineal. Mientras que el SVM es un clasificador binario, el rendimiento de los árboles de decisión puede mejorarse mediante ensembles entrenados en conjuntos binarios obtenidos a partir del original ("ensembles de binarización").
- En el grupo de clasificadores *general*, se usaron diversas técnicas entre las más populares en la literatura de Minería de Datos. Estos clasificadores fueron: redes neuronales (Multi-Layer Perceptron, MLP) , Naïve Bayes, k vecinos, árboles de decisión C4.5 y tres ensembles de árboles: Bagging, Adaboost y Rotation Forest.
- El grupo de clasificadores *binarios-desequilibrado* está formado por SMOTEBoost, RusBoost y SMOTEBagging. Esos métodos no se combinaron con técnicas de remuestreo, ya que las realizan internamente.

A partir del conjunto de datos inicial se formaron diferentes conjuntos con varios grados de desequilibrio de la clase minoritaria. Así, se varió el número de instancias de las clases minoritarias (aquellas que representan los fallos en los aerogeneradores) del 3 al 12,5 % del total del número de instancias. Este rango de desequilibrio busca simular las condiciones reales de conjuntos de datos en un contexto de producción. De hecho, se fijó el límite del 3 % debido a que algunas técnicas que utilizan undersampling no podrían funcionar por debajo de ese límite (debido a un insuficiente número de instancias de algunas clases).

Tras analizar los 8 grupos de experimentos, se seleccionaron las dos mejores técnicas de clasificación de cada grupo: el clasificador que maximiza MacroF con menor y mayor grado de desequilibrio (si un clasificador es el mejor en ambas situaciones, sólo se selecciona un clasificador para ese grupo de técnicas). Los resultados de la selección fueron:

- Entre clasificadores con undersampling: Rotation Forest de C4.5.
- Entre clasificadores con oversampling: Rotation Forest de C4.4 y Adaboost de C4.4.
- Entre ensembles de binarización de clasificadores binarios no orientados a problemas desequilibrados: ECOC con C4.4 y ECOC con SVM lineal.
- Entre ensembles de binarización de clasificadores binarios orientados a problemas desequilibrados: ECOC con SMOTEBoosting de C4.4 y ECOC con RUSBoost de C4.4.
- Entre ensembles de binarización con undersampling: ECOC con C4.4 y ECOC con C4.5.
- Entre ensembles de binarización con oversampling: ECOC con C4.5 y ECOC con SVM lineal.
- Entre clasificadores sensibles al coste: Rotation Forest de C4.4.
- Entre clasificadores de uso general: Adaboost de C4.4 y Adaboost de C4.5.

Una vez que se disponía de los mejores clasificadores, se analizó la sensibilidad de las distintas técnicas al grado de desequilibrio. En el Cuadro 5.10 se muestra cómo se ve afectado el nivel de MacroF alcanzado por el grado de desequilibrio, indicando el número de clasificadores que alcanzan un valor de MacroF de 0,5 a cada nivel. Este análisis muestra que un ensemble Rotation Forest con árboles de decisión C4.4 como clasificadores base es el más robusto al grado de desequilibrio, puesto que su MacroF no bajó de 0,5.

En la Figura 5.1 se comparan los valores de MacroF para los mejores métodos. Como muestra la figura, un clasificador sensible al coste con un ensemble Rotation Forest de árboles C4.4 de clasificadores base fue el mejor método encontrado en la experimentación en términos de la métrica MacroF. Para dicho método se obtuvo el mejor valor de la métrica para todos los porcentajes de las clases minoritarias.

Cuadro 5.10: Sensibilidad al grado de desiquilibrio de las diferentes estrategias

Porcentaje de las clases minoritarias	Número de métodos alcanzando $MacroF \geq 0{,}5$
<4 %	Sólo 1
4-4,5 %	6
>5,5 %	Todos (14)

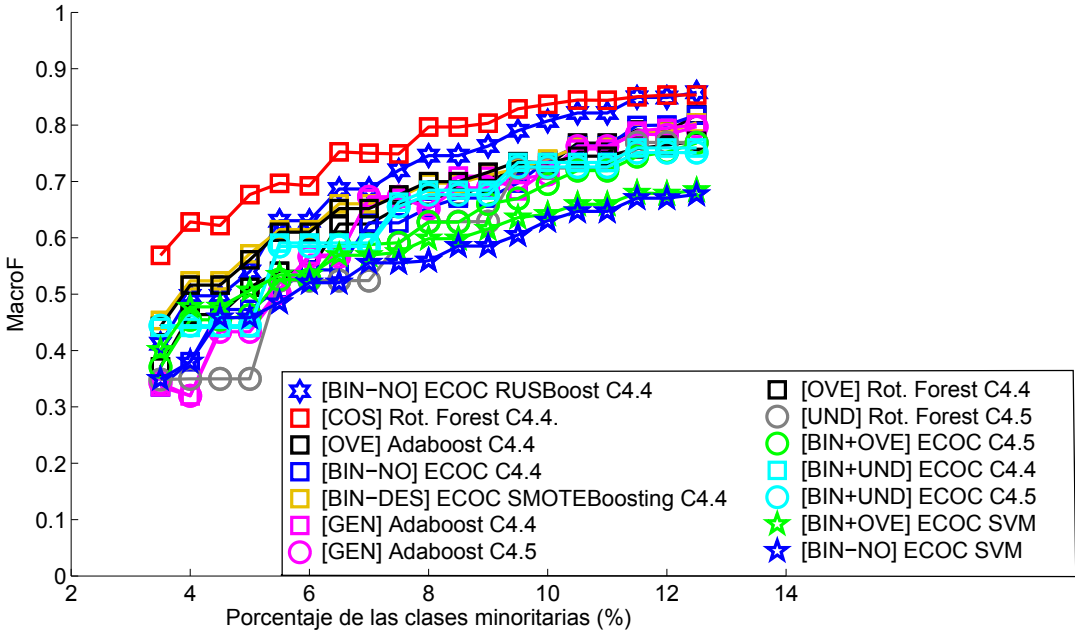


Figura 5.1: MacroF – Mejores métodos

Se extrajeron las siguientes conclusiones para los mejores métodos en términos de la métrica MacroF:

- Las técnicas de clasificación sensibles al coste y los ensembles de binarización con clasificadores específicos de problemas binarios desequilibrados fueron mejores estrategias para el problema de clasificación multiclase desequilibrado que otras estrategias como usar técnicas de remuestreo fuera de los clasificadores o clasificadores sin ninguna técnica específica de problemas multiclase.

- Entre los mejores ensembles, C4.4 fue un mejor clasificador base que C4.5.

- Entre los ensembles de binarización, el sistema de combinación ECOC fue mejor que los sistemas 1 contra 1 y 1 contra todos.

- Hubo dos casos para los que se obtuvo un valor muy bajo de la métrica MacroF. Esto pudo deberse a un problema de sobre-ajuste. Estos casos fueron un ensemble de binarización con árboles C4.5. (0,030 para un porcentaje de las clases minoritarias en el rango 3,5-6 %) y una red neuronal dentro de un clasificador sensible al coste (sobre 0,028).

- Cuando el porcentaje de las clases minoritarias es superior al 10 %, la pendiente de subida de las curvas se suaviza (i.e., el rendimiento de los métodos se estabiliza). Esto significa que si el conjunto de datos incluye un alto número de instancias (algo difícil a nivel industrial) las diferencias de rendimiento entre clasificadores son menos importantes.

- Cuando el grado de desequilibrio es muy alto se tienen valores pobres de la métrica MacroF. Considerando que 0,5 es el valor estándar para una predicción aleatoria, se fijó una referencia mínima de 0,75 para analizar el máximo grado de desequilibrio con el que los clasificadores pueden obtener precisiones aceptables. El mejor método (clasificador sensible al coste con Rotation Forest de C4.4) alcanzó esta referencia cuando el porcentaje de las clases minoritarias fue el 6,5 %, mientras que el segundo método (ensemble de binarización ECOC con Rusboost de C4.4) necesitó que el porcentaje ascendiera al 8 %. Para el resto de clasificadores, no se alcanzó la referencia hasta que el porcentaje de las clases minoritarias subió hasta 10,5 %.

- La métrica MacroF tomó un valor en torno a 0,85 al final de las curvas. Esto puede considerarse un buen rendimiento, dado que el problema tiene ocho clases y aún para ese nivel sigue siendo muy desequilibrado.

Wind turbines fault diagnosis using ensemble classifiers

Pedro Santos[1], Luisa F. Villa[2], Aníbal Reñones[2], Andrés Bustillo[1], Jesús Maudes[1]

[1] Department of Civil Engineering, University of Bugos
C/ Francisco de Vitoria s/n, 09006, Burgos, Spain
{psgonzalez,abustillo,jmaudes}@ubu.es
[2] CARTIF Foundation
Parque Tecnológico de Boecillo, 47151 Boecillo, Valladolid, Spain
{luivil,aniren}@cartif.es

Abstract. Fault diagnosis in machines that work under a wide range of speeds and loads is currently an active area of research. Wind turbines are one of the most recent examples of these machines in industry. Conventional vibration analysis applied to machines throughout their operation is of limited utility when the speed variation is too high. This work proposes an alternative methodology for fault diagnosis in machines: the combination of angular resampling techniques for vibration signal processing and the use of data mining techniques for the classification of the operational state of wind turbines. The methodology has been validated over a test-bed with a large variation of speeds and loads which simulates, on a smaller scale, the real conditions of wind turbines. Over this test-bed two of the most common typologies of faults in wind turbines have been generated: imbalance and misalignment. Several data mining techniques have been used to analyze the dataset obtained by order analysis, having previously processed signals with angular resampling technique. Specifically, the methods used are ensemble classifiers built with *Bagging*, *Adaboost*, *Geneneral Boosting Projection* and *Rotation Forest*; the best results having been achieved with *Adaboost* using C4.5 decision trees as base classifiers.

Keywords: fault diagnosis, wind turbines, ensemble classifiers, angular resampling

1 Introduction

Vibration analysis has been studied and applied to rotating machinery for decades. It is widely accepted as one of the main fault diagnosis techniques in machine maintenance [11]. As the signal analysis technology has advanced and new sensors have been developed, fault diagnosis and maintenance of machines working under more severe conditions have

become a new target area for experts. Examples of machines that work under variable conditions of load and speed are wind turbines, excavators and helicopters [2]; [4]; [5]; [3]. Gear transmission plays a crucial role in the reliability of these machines.

One of the first research in the field of transmission damage diagnosis focused on vibration signals analysis [6]. At first, the statistical features of the signal in the time domain were the main element of study [18]. However, the field quickly spread to include spectral analysis, time-frequency analysis and, finally, models based on artificial intelligence. All of these approaches are still valid and current. As new techniques of signal processing arise, they are applied to the problem of damage detection in chain drives and should be adapted to the needs and specific characteristics of each mechanical system.

The main purpose of this work is to study fault diagnosis in wind turbines. To do so, the test-bed shown in Figure 1 is used to approximate real conditions and the typical faults of a real wind turbine.

Many studies have applied several signal analysis methods that are suited to conditions of fluctuating loads. Among these, we may quote works by Stander, Heyns, Zhan and Barlelmus [21]; [24]; [3]. However, no studies have yet been completed on such wide working ranges as those of a wind turbine, in terms of real wind regimes that therefore have a very wide range of speed and load operating conditions. The development of intelligent devices, both for monitoring and for diagnosis of this type of industrial equipment that operates under highly variable loads and speeds is, therefore, a highly topical field of research. Vibration monitoring systems require signal processing procedures to compensate for fluctuations in axis speeds and amplitude modulation, due to the variable wind-resistance loads [20]; [19].

Although exhaustive research into the analysis of the signals obtained from several types of sensors and particularly accelerometers has been completed to date, the standard technique used for fault diagnosis is the identification of critical variables by an expert and the development of a regression model that forecasts the failure [24]. The aim of this work is to develop an alternative classification system with greater reliability using ensemble classifiers.

There are several works in which ensemble classifiers have been used for fault detection. In Hu [12] *Adaboost* is used to combine *Support Vector Machines* (a type of base classifier) for fault diagnosis in ro-

tating machinery. This method is also used in Donat [8] for the fault detection of engines in gas turbines. In Alonso [1] , failure identification in continuous processes is managed by an ensemble classifier building method -*Stacking*- that combines nearest-neighbours base classifiers (*k-Neighbours Classifier, kNN*). Furthermore, *Adaboost* and *Bagging* of neural networks in El-Gamal [9] are used for fault diagnosis in analogue circuits.

2 Description of the test-bed and measurement procedure

The experiments conducted on the test-bed are meant to simulate the behaviour of wind turbines. This test-bed is used to simulate different defects under variable loads and speeds. The right side of the test-bed (Figure 1) is composed of an engine, a parallel gearbox and a planetary gearbox. Both gearboxes resemble a commercial wind turbine in terms of their configuration and gear ratios (1:61).

To simulate the variable load in the drive train of a wind turbine, due to randomness of the wind, an electric brake has been added to the right side of the bench.

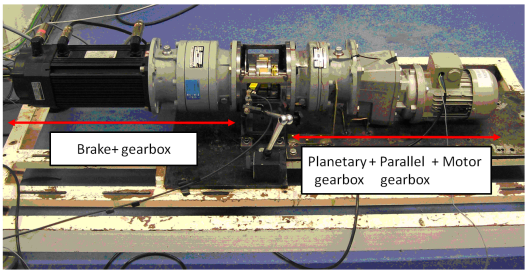


Fig. 1. Figure 1: Test-Bed

For the measurement of vibration signals four accelerometers distributed in the axial and radial position in the gearboxes situated on the right side of the test-bed were used.

Preliminary processing of the vibration signals need to be performed, due to the speed and load variations caused by the operating conditions of

wind turbines, in order to extract the information on its spectral analysis. The technique of angular sampling, a methodology that may be found in [22], appears suitable to solve this problem.

The faults simulated on the test-bed were imbalance and misalignment, starting with small values and increasing at each measurement to simulate a progressive fault (Table 1). This table illustrates the value of the weight in grams and its equivalent in percentages with regard to the total of the weight of the rotor of the bench, and the thickness used for producing the misalignment, as well as the resulting value.

Table 1. Types of faults and magnitudes induced in the test-bed

Imbalance			**Misalignment**		
	gr	%		mm	°
Imbalance A	5.79	0.077	Misalignment A	0.75	1.53
Imbalance B	9.13	0.12	Misalignment B	0.75	1.53
Imbalance C	19.5	0.26			
Imbalance D	28.8	0.38			

To guarantee the speed and load conditions, several profiles were generated to cover a wide range of speeds from 1000 to 1800 rpm at random, and from 0 to 100 % of the load. An example of this profile is shown below in the Figure 2.

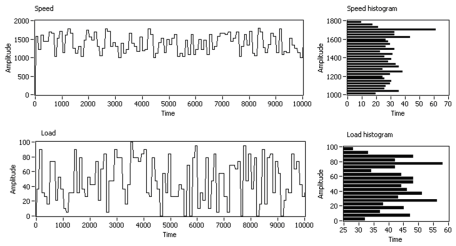


Fig. 2. Figure 2: Speed and load profile

These profiles were generated to cover a whole day of measurement (24 hours), with constant 100 second intervals of speed and load. The speed measurements were generated from 1000 rpm, which is the ap-

proximate speed at which a wind turbine begins to produce energy. Data acquisition was taken at intervals of 72 seconds from each of the four accelerometers with a sampling frequency of 25600 Hz. The speed signal was captured at 6400 Hz.

The set of tests done are reported in [23].

3 Variables analyzed

As explained in the previous section, several working faults in the turbine are analyzed. For that reason, a discrete output variable was defined, referred to as the fault type, and several input variables.

The type of fault matches the two previously explained ones; misalignment and imbalance, for which there are three possible numerical values in the first case (0, 0.75 and 2) and five in the second one (0, 5.79, 9.13, 19.5 and 28.8). We will refer to these degrees of misalignment as DA1, DA2 and DA3, and to imbalance as DB1, DB2, DB3 DB4, DB5.

There are therefore fifteen possible values for each type of faults, shown below in Table 2:

Table 2. Fault Classes

	DB1	**DB2**	**DB3**	**DB4**	**DB5**
DA1	0 (C0)	1 (C1)	2 (C2)	3 (C3)	4 (C4)
DA2	5 (C5)	6	7	8	9
DA3	10 (C6)	11	12	13	14 (C7)

In the previous table, class 0 matches the case in which there is no fault (no misalignment nor imbalance), and the 14 remaining classes match several types of faults that could theoretically occur, but in the experimental trials only 8 classes took place. These fault classes will be referred to as C0, C1, C2, C3, C4, C5, C6 and C7.

The variables in this problem are, on the one hand, 3 magnitudes which describe the operational state of the machine in the terms of torque, speed and electric input current and, on the other, several magnitudes measured with 5 sensors, 1 current sensor and 4 accelerometers, 2 by each of the two gearboxes, distributed along two perpendicular axis.

The current sensor provides 4 measurements of electric current, and the accelerometers provide the data for a vibration analysis along the

axis, by using three aspects of the vibration spectrum. On the one hand, 5 measurements which summarize their distribution (average, RMS, skewness, kurtosis and interquartile range); on the other, a harmonic analysis (natural frequency of system vibrations and multiples thereof, 80 measurements in total); and finally, dividing the vibration spectrum into bands of fixed position (unrelated to the natural frequency of the system), with another 77 measurements. Each accelerometer provides a total of 162 measurements, although the total number of considered variables in the vibration analysis is 537, as some measurements with redundant information have been removed.

The final number of variables for the problem is 544, adding to the 537 from the vibration analysis, the measurements from the current sensor, the torque, the speed and the electric current. In the next table, a summary of the previously explained variables is completed, although it is possible to search for a more detailed information in [23].

Table 3. Input variables

Operation State		
Variable	**Number of measurments**	**Units**
Torque	1	% of maximum torque
Speed	1	Hz
Input current	1	Amperes
Current Sensors		
Variable	**Number of measurments**	**Units**
Electrical current in the axis	4	Amperes
Vibration analysis		
Variable	**Number of measurments**	**Units**
Harmonics	272	mm/s^2
Bands	245	mm/s^2
Average	4	mm/s^2
RMS	4	mm/s^2
Skewness	4	dimensionless
Kurtosis	4	dimensionless
Interquartile Range	4	mm/s^2

During the day of the experimentation, 6551 different conditions in the considered variables were registered. The data set under study therefore has a size of 6551 instances with 544 attributes, such that it can be considered a high dimensional problem.

The distribution of the instances among the classes is as shown in Table 4:

Table 4. Distribution of the instances among the classes

C0	887 (13.54 %)
C1	847 (12.93 %)
C2	856 (13.07 %)
C3	838 (12.79 %)
C4	864 (13.19 %)
C5	872 (13.31 %)
C6	835 (12.75 %)
C7	552 (8.43 %)

4 Fault analysis by ensemble classifiers

Forecasting several faults that may occur in turbine operation is included in data mining classification problems. In this article, the use of techniques to combine several individual classifiers is proposed, to obtain an ensemble classifier. These techniques have developed over the last decade and their output has been proven in several situations.

An ensemble classifier is a classification technique by which the forecasted class is obtained from the individual forecasts of a series of base classifiers. There are several ways of combining the various forecasts, the most usual one is to select the most voted class. The global accuracy of the ensemble classifier depends on the diversity of the classifiers and their individual accuracy, as an ensemble classifier should be capable outperforming any individual classifier [7]; [14].

There are several ways of forcing diversity between base classifiers [13]; [17], having taken four of these techniques in this study, *Bagging* and *Adaboost* on the one hand, are the most commonly used, and *Rotation Forest* and *General Boosting Projection (GBPC)* on the other, which are more novel techniques that have been shown to be very competitive [16]; [10].

The algorithm *Rotation Forest* algorithm is based on Principal Component Analysis (PCA) extraction procedures that achieve better accuracy in the ensemble classifier, by acting at the same time on the individual accuracy of each base classifier and on its diversity [16]. Thus,

a random division of the data is made, in groups of attributes (3 in this work), and subsequently a PCA analysis is completed over part of the samples of each group, also random, storing the projection matrix that is used and combined later on to project all the samples of each group.

The *GBPC (General Boosting Projection)* is based on the use of supervised projections to improve global accuracy, due to the individual improvement of each base classifier as well as its diversity [10]. It is an iterative process in which the first base classifier receives the data set without any modification followed by a projection over the misclassified instances by the previous classifier. By doing so, we seek to obtain better results in the next classifier, in cases where the previous classifiers failed. The Non-parametric Discriminant Analysis (NDA) version proposed by [15] was used as the supervised projection method.

5 Results

Three methodologies for the classification were tested: C4.5 decision trees, k-Nearest Neighbour (*kNN*) and Naive Bayes. These three base techniques were chosen as they are the three most commonly used in data mining.

These methods have been tested individually as well as with ensemble classifiers using the techniques of *Bagging*, *Adaboost*, *GBPC* and *Rotation Forest*, taking in all cases 100 base classifiers, and performing a 5×2 cross validation (all the methods are compared using the same sets for training and testing).

Two ways to measure the accuracy of each classifier have been taken:

- Success rate in a 5×2 cross validation, indicating the standard deviation of the iterations.
- Confusion matrix, in which the class forecasted by the classifier is compared against the class of the instance to which actually belongs.

5.1 Success rate

The following table illustrates the success rate of both the individual and the different ensemble classifiers, which includes the standard deviation with regard to the 5 repetitions of the cross validation between parentheses.

In all cases, we can see that decision trees are more suitable as base classifiers, however we should highlight the notable increase of the *GBPC* with regard to the efficiency of the classification with the *kNN* as base classifier.

Table 5. Average success and standard deviation for the different classifiers

	C4.5 trees	**kNN**	**Naive Bayes**
Classifier individually	92.60 (0.51)	66.12 (0.44)	70.29 (2.68)
Bagging	95.33 (0.23)	66.01 (0.41)	70.73 (1.59)
Adaboost	96.24 (0.12)	67.57 (0.59)	78.96 (0.71)
GBPC (NDA)	90.70 (5.61)	87.45 (1.26)	70.29 (2.68)
Rotation forest	95.84 (0.14)	66.19 (0.54)	71.92 (2.25)

The low performance of the kNN classifier could be caused by the well-known problem of the "curse of dimensionality" (analyzing high-dimensional spaces). In the following sections we compare the two methods in which better results are reached, *Adaboost* with decision trees versus *Rotation Forest* with decision trees.

5.2 Confusion Matrix

The next step is to compare the results of *Adaboost* and *Rotation Forest* with 100 C4.5 trees as base classifiers, by using the average confusion matrix of the 5×2 ccross validation (the confusion matrix average of those provided by each of the 10 classifiers obtained in the cross validation has been calculated, and the values have been rescaled with regard to the total).

Regarding to the operation control, the most critical cases are those registered in the first column in both tables, as they match with those in which the ensemble classifier estimates that there are not a fault operation. By analyzing the data of this column, we can see that the undetected percentage of errors is 0.23 % in the case of *Adaboost*, and 0.68 % in the case of *Rotation Forest*.

Using the t test to compare the ensemble classifiers *Adaboost* and *Rotation forest* with a level of significance of 1 %, we may conclude from the statistical evidence that the first algorithm outperforms the second one with regard to the way it models the data.

Table 6. Confusion matrix for *Adaboost* (top) / *Rotation Forest* (bottom) of C4.5 trees

	C0	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**
C0	13.40	0.00	0.14	0.00	0.00	0.00	0.00	0.00
C1	0.00	11.53	0.03	1.18	0.19	0.00	0.00	0.00
C2	0.01	0.00	12.87	0.19	0.00	0.00	0.00	0.00
C3	0.02	0.56	0.27	11.23	0.71	0.00	0.00	0.00
C4	0.01	0.04	0.00	0.41	12.73	0.00	0.00	0.00
C5	0.00	0.00	0.00	0.00	0.00	13.31	0.00	0.00
C6	0.00	0.00	0.00	0.00	0.00	0.00	12.75	0.00
C7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.42

	C0	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**
C0	13.39	0.00	0.15	0.00	0.00	0.00	0.00	0.00
C1	0.00	11.80	0.03	0.90	0.20	0.00	0.00	0.00
C2	0.03	0.00	12.88	0.15	0.00	0.00	0.00	0.00
C3	0.02	0.78	0.48	10.72	0.79	0.00	0.00	0.00
C4	0.04	0.09	0.02	0.46	12.58	0.00	0.00	0.00
C5	0.00	0.00	0.00	0.00	0.00	13.31	0.00	0.00
C6	0.00	0.00	0.00	0.00	0.00	0.02	12.73	0.00
C7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.43

6 Conclusions

This study has proposed a fault diagnosis system for machines with high variation in the speed and load conditions, such as wind turbines. These devices have undergone significant growth over the last five years and require immediate industrial solutions to their tele-maintenance problems. The failure diagnosis system explained in this work consists of several measurement sensors, especially accelerometers, signal analysis equipment based on resampling angular techniques to process the data from these sensors, and a module that implements different data mining techniques for the classification of the operational state of wind turbines. Several methods of combining base classifiers have been applied to identify seven different levels of two typical faults in wind turbines: misalignment and imbalance. *Adaboost* using J48 decision trees as base classifiers achieved high accuracy (correct forecasts in 96.24 % of cases) when analyzing a wide real dataset measured on a test-bed that simulate real conditions of wind turbines operation (65551 instances with 544 attributes). Future research will be focused in the improvement of the industrial application through the testing of the proposed fault diagnosis

system on a more extensive dataset that includes more fault cases and has been recorded under real industrial conditions, because the analysed dataset reflects a limited number cases of two fault types (misalignment and imbalance).

7 Aknowledgments

This work has been partially funded by the Ministry of Science and Innovation of Spain through the MAGNO project (Ref. 2008/1028), within the CENIT funding programme.

References

1. Alonso, C.J., Prieto, O.J., Rodríguez, J.J., Bregón, A., Pulido, B.: Current topics in artificial intelligence. chap. Stacking Dynamic Time Warping for the Diagnosis of Dynamic Systems, pp. 11–20. Springer-Verlag, Berlin, Heidelberg (2007), http://dx.doi.org/10.1007/978-3-540-75271-4_2
2. Barszcz, T., Randall, R.B.: Application of spectral kurtosis for detection of a tooth crack in the planetary gear of a wind turbine. Mechanical Systems and Signal Processing 23(4), 1352–1365 (2009), http://www.sciencedirect.com/science/article/pii/S0888327008002239
3. Bartelmus, W., Zimroz, R.: Vibration condition monitoring of planetary gearbox under varying external load. Mechanical Systems and Signal Processing 23(1), 246–257 (2009), http://www.sciencedirect.com/science/article/pii/S0888327008000824, special Issue: Non-linear Structural Dynamics
4. Blunt, D.M., Keller, J.A.: Detection of a fatigue crack in a uh-60a planet gear carrier using vibration analysis. Mechanical Systems and Signal Processing 20(8), 2095–2111 (2006), http://www.sciencedirect.com/science/article/pii/S0888327006001245
5. Combet, F., Zimroz, R.: A new method for the estimation of the instantaneous speed relative fluctuation in a vibration signal based on the short time scale transform. Mechanical Systems and Signal Processing 23(4), 1382–1397 (2009)
6. Davies, A.: Handbook of condition monitoring: techniques and methodology. Chapman & Hall (1998), http://books.google.es/books?id=j2mN2aIs2YIC
7. Dietterich, T.: Ensemble methods in machine learning. In: Multiple Classifier Systems, Lecture Notes in Computer Science, vol. 1857, pp. 1–15. Springer Berlin / Heidelberg (2000), http://dx.doi.org/10.1007/3-540-45014-9_1, 10.1007/3-540-45014-9_1
8. Donat, W., Choi, K., An, W., Singh, S., Pattipati, K.: Data visualization, data reduction and classifier fusion for intelligent fault diagnosis in gas turbine engines. Journal of Engineering for Gas Turbines and Power 130(4), 041602 (2008), http://link.aip.org/link/?GTP/130/041602/1
9. El-Gamal, M., Mohamed, M.: Ensembles of neural networks for fault diagnosis in analog circuits. Journal of Electronic Testing 23, 323–339 (2007), http://dx.doi.org/10.1007/s10836-006-0710-1, 10.1007/s10836-006-0710-1
10. García-Pedrajas, N., García-Osorio, C.: Constructing ensembles of classifiers using supervised projection methods based on misclassified instances. Expert Syst. Appl. 38(1), 343–359 (2011)

11. Hameed, Z., Hong, Y., Cho, Y., Ahn, S., Song, C.: Condition monitoring and fault detection of wind turbines and related algorithms: A review. Renewable and Sustainable energy reviews 13(1), 1–39 (2009)
12. Hu, Q., He, Z., Zhang, Z., Zi, Y.: Fault diagnosis of rotating machinery based on improved wavelet package transform and svms ensemble. Mechanical Systems and Signal Processing 21(2), 688–705 (2007), http://www.sciencedirect.com/science/article/pii/S0888327006000306
13. Kuncheva, L.: Combining classifiers: Soft computing solutions. Pattern Recognition: From Classical to Modern Approaches pp. 427–451 (2001)
14. Kuncheva, L.: Combining pattern classifiers: methods and algorithms. Wiley-Interscience (2004), http://books.google.es/books?id=9TJ6igZtqWAC
15. Kuo, B., Ko, L., Pai, C., Landgrebe, D.: Regularized feature extractions for hyperspectral data classification. In: Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International. vol. 3, pp. 1767–1769. IEEE (2003)
16. Rodriguez, J., Kuncheva, L., Alonso, C.: Rotation forest: A new classifier ensemble method. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28(10), 1619–1630 (oct 2006)
17. Rokach, L.: Ensemble-based classifiers. Artificial Intelligence Review 33, 1–39 (2010), http://dx.doi.org/10.1007/s10462-009-9124-7, 10.1007/s10462-009-9124-7
18. Samuel, P.D., Pines, D.J.: A review of vibration-based techniques for helicopter transmission diagnostics. Journal of Sound and Vibration 282(1-2), 475–508 (2005), http://www.sciencedirect.com/science/article/pii/S0022460X04003244
19. Stander, C.J., Heyns, P.S.: Instantaneous angular speed monitoring of gearboxes under non-cyclic stationary load conditions. Mechanical Systems and Signal Processing 19(4), 817–835 (2005), http://www.sciencedirect.com/science/article/pii/S0888327004001633
20. Stander, C.J., Heyns, P., Schoombie, W.: Using vibration monitoring for local fault detection on gears operating under fluctuating load conditions. Mechanical Systems and Signal Processing 16(6), 1005–1024 (2002), http://www.sciencedirect.com/science/article/pii/S0888327002914792
21. Stander, C., Heyns, P.: Transmission path phase compensation for gear monitoring under fluctuating load conditions. Mechanical Systems and Signal Processing 20(7), 1511–1522 (2006), http://www.sciencedirect.com/science/article/pii/S0888327005000919
22. Villa, L.F., Renones, A., Perán, J.R., de Miguel, L.J.: Angular resampling for vibration analysis in wind turbines under non-linear speed fluctuation. Mechanical Systems and Signal Processing 25(6), 2157–2168 (2011), http://www.sciencedirect.com/science/article/pii/S0888327011000677, interdisciplinary Aspects of Vehicle Dynamics
23. Villa, L.F., Renones, A., Perán, J.R., de Miguel, L.J.: Statistical fault diagnosis based on vibration analysis for gear test-bench under non-stationary conditions of speed and load. Mechanical Systems and Signal Processing. In Press, Accepted Manuscrit, doi: 10.1016/j.ymssp.2011.12.013 (2012)
24. Zhan, Y., Makis, V., Jardine, A.K.: Adaptive state detection of gearboxes under varying load conditions based on parametric modelling. Mechanical Systems and Signal Processing 20(1), 188–221 (2006), http://www.sciencedirect.com/science/article/pii/S0888327004001499

Sensors **2015**, *15*, 5627-5648; doi:10.3390/s150305627

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

An SVM-Based Solution for Fault Detection in Wind Turbines

Pedro Santos [1], Luisa F. Villa [2], Aníbal Reñones [2], Andres Bustillo [1,*] and Jesús Maudes [1]

[1] Department of Civil Engineering, University of Burgos, C/ Francisco de Vitoria s/n, Burgos 09006, Spain; E-Mails: psgonzalez@ubu.es (P.S.); jmaudes@ubu.es (J.M.)
[2] CARTIF Foundation, Parque Tecnológico de Boecillo, Boecillo 47151, Spain; E-Mails: lvillamo@eafit.edu.co (L.F.V.); aniren@cartif.es (A.R.)

* Author to whom correspondence should be addressed; E-Mail: abustillo@ubu.es; Tel.: +34-947-259358; Fax: +34-901-706775.

Academic Editor: Vittorio M.N. Passaro

Received: 22 January 2015 / Accepted: 25 February 2015 / Published: 9 March 2015

Abstract: Research into fault diagnosis in machines with a wide range of variable loads and speeds, such as wind turbines, is of great industrial interest. Analysis of the power signals emitted by wind turbines for the diagnosis of mechanical faults in their mechanical transmission chain is insufficient. A successful diagnosis requires the inclusion of accelerometers to evaluate vibrations. This work presents a multi-sensory system for fault diagnosis in wind turbines, combined with a data-mining solution for the classification of the operational state of the turbine. The selected sensors are accelerometers, in which vibration signals are processed using angular resampling techniques and electrical, torque and speed measurements. Support vector machines (SVMs) are selected for the classification task, including two traditional and two promising new kernels. This multi-sensory system has been validated on a test-bed that simulates the real conditions of wind turbines with two fault typologies: misalignment and imbalance. Comparison of SVM performance with the results of artificial neural networks (ANNs) shows that linear kernel SVM outperforms other kernels and ANNs in terms of accuracy, training and tuning times. The suitability and superior performance of linear SVM is also experimentally analyzed, to conclude that this data acquisition technique generates linearly separable datasets.

Keywords: fault diagnosis; neural networks; support vector machines; wind turbines

1. Introduction

Over the last decade, the exponential growth of wind farms around the world has involved significant challenges, in order to improve their efficiency [1] and to reduce their operational costs [2]. The reduction of maintenance costs is the main strategy to reduce operational costs, because wind turbine maintenance is always a complex and expensive task, due to difficulties over their positioning and their distance from industrial areas. The recent development of offshore wind farms adds a further dimension to this industrial issue. The development and integration of on-line intelligent multisensor systems for the monitoring and diagnosis of wind turbines appears to be the best strategy to reduce their maintenance costs. However, an intelligent system, able to cover all possible failures, from electrical to mechanical, is still too great a problem to grapple with nowadays. For this reason, any monitoring system of wind turbines should focus on a wind turbine sub-assembly: the mechanical chain, the power generator, the lubricating system, *etc*. Once the sub-assembly is chosen, some steps should be followed: first, the most suitable sensors and their position in the sub-assembly should be chosen; second, the most suitable data-processing techniques should be found for each kind of signal collected from the sensors; third the best data-mining technique to extract as much information as possible from the processed signals should be identified. This Introduction addresses all these questions for a specific wind-turbine maintenance problem: the mechanical chain.

A variety of reviews on wind farm maintenance have been published [2–4]. Joselin *et al.* [2] include the power chain or gearbox and the rotor blade in the list of the more complex components in the performance of a wind turbine, and Tavner *et al.* [5] present these two elements as responsible for a significant number of wind turbine failures. The most common failures of these components are rotor blade misalignment and imbalance of the power chain due to bearing fatigue and gear damage [2]. Furthermore, the downtime due to such failures is even more important, due to procurement times and the need to winch these heavy sub-assemblies in and out of the nacelle.

There are different kinds of sensors that could be used for fault diagnosis of the gearbox and the generator subassemblies in wind turbines. Analysis of the power signals emitted by wind turbines is insufficient for this diagnosis. The limitations for the use of power signals in the mechanical diagnosis have their origin in the low-pass filter nature of the electrical motor itself. These power signals can be useful to detect concrete defects (of low frequency) in the generator, such as imbalance, but fail to detect the early stage of others defects, such as bearings (at first, of high frequency). In addition to that, due to the mechanical coupling between the generator and the rest of the power-train (gearbox and main shaft), the diagnosis of defects in those elements via power signals is even more difficult, if not impossible. Therefore, most of the sensors used for fault diagnosis in these devices attempt to evaluate vibrations generated during the rotation of these components. Accelerometers are therefore the most common sensors for this task. Vibration analysis has been studied and applied to rotating machinery for decades. It is widely accepted as one of the main fault diagnosis techniques in machine maintenance [6]. However, the case of wind turbines presents a new challenge in comparison with other rotatory machinery [7,8]: these devices work under variable conditions of load and speed. Only a few machines, such as excavators and helicopters, present these characteristics [9]. Therefore, the information obtained from accelerometers should be completed with information from other sensors

that monitor the real-time rotating speed or the efficiency of the energy generation process, to locate the origin of the failure, and a multisensor approach is often proposed in the bibliography as a better solution to these industrial tasks [4].

Once the most suitable sensors have been selected, different approaches to data analysis may be chosen for failure detection in wind turbines. Spectral analysis of the power output signal has been proposed to monitor rotor blade imbalance [10] and gearbox and bearing faults [11]. However, in general, vibration sensors have been favored [12]. At first, the statistical features of the vibration signal in the time domain were studied [13], although this field of study quickly spread to include spectral analysis and time-frequency analysis [14].

Once signals from the sensors have been acquired, different techniques can be used to extract as much information as possible from these data, so as to build-up a suitable decision-making system for failure detection in wind turbines. Previous studies have applied different data-mining techniques to this industrial task, such as SVM [15,16], Bayesian networks [17], self-organizing maps [18], ANNs [19], ensemble classifiers [20] and neuro-fuzzy inference systems [16]. Moreover, recent works have proven the suitability of AI techniques for other similar industrial tasks, such as evaluation of the mechanical properties in rapid prototyping using ensembles [21], tool condition monitoring using ν-SVM [22] and ensembles of SVMs, hidden Markov model and the radius basis function [23].

SVMs with different kernels were tested on the dataset, to identify the quickest and the most reliable and accurate technique for this industrial task. ANNs were also tested as a standard technique to compare the results, as described in the Bibliography [13]. However, the use of ANNs has some disadvantages, as their results are highly dependent on the optimization of their tuning parameters, and the process of fine-tuning these parameters requires a lot of work and experience [24]. There are no general rules that may be followed as a guide, turning the scientific process into somewhat of an art. In contrast, SVM techniques are easier to use and have been successfully applied to many classification problems, such as recognition of isolated digits [25], objects [26], faces [27], speaker identification [28] and text categorization [29].

Even though extensive research has been completed into fault diagnosis in wind turbines, the standard technique used nowadays for this industrial task under real working conditions is the identification of critical variables by an expert and the development of a regression model that predicts the failure [14]. This solution requires in-depth knowledge of the mechanical chain and its components in such devices, especially to identify critical frequencies to detect imbalance failures. It has been adopted because existing studies are not suitable for real industrial conditions, as they follow two approaches with severe limitations. The first approach chose accelerometers as sensors and high frequency acquisition rates (kHz) to diagnosis the wind turbines. In this case, however, the recorded periods of time (in the order of months) are not enough to provide a sufficiently accurate failure map. Neither do these studies cover the wide working ranges of a wind turbine under real wind regimes, which necessarily involve operating conditions with very different speeds and loads. The second approach is found in SCADA systems integrated in the controls of real wind farms; these acquisition systems have no accelerometers, because their acquisition rates are low (Hz), and their predictive capabilities are therefore very limited. The novelty of this work is combining an extensive range of working conditions, obtained from a test-bed, with different failure levels for two of the most common failures in the mechanical chain of a wind

turbine. In addition to this novel aspect, our study proposes a holistic strategy to build an optimized decision-making scheme to detect failures in such devices; Figure 1. Firstly, different sensors, including accelerometers, are setup in the wind turbine simulator, in positions that are also attainable in real wind turbines, and an extensive experimental program under random conditions is performed; secondly, a suitable technique for data processing of vibration signals at various rotation speeds is implemented; thirdly, different data-mining techniques, SVMs with different kernels, are tested on the dataset to identify the quickest, most accurate and reliable technique for this industrial task. ANNs were also tested, as they are a standard technique in the Bibliography, to compare the results [13]. This methodology also presents a further advantage regarding its implementation in real wind turbines; its configuration neither requires information on the mechanical design, nor on the presence of an expert, because the entire frequency spectrum is systematically analyzed (considering theoretical vibrational levels, to define all possible characteristic frequencies where faults may generate vibrations). The result of that analysis is a dataset containing sufficient representative instances for normal and failure working states. These instances also have a large number of features that help the machine learning method to choose the most suitable one for each wind turbine design. A very simple machine learning method, such as a linear SVM, with this specific dataset structure, outperforms some other sophisticated alternatives, as shown in the experiments. Similar schemes have been successfully applied to other industrial tasks, such as breakage detection [30,31] and roughness prediction [32]. In a previous study, we analyzed this dataset using ensemble classifiers [20], as this technique has been successfully applied to other similar industrial problems [24,32]. However, in this new work, we have obtained better statistically-significant results and a more reliable diagnosis system.

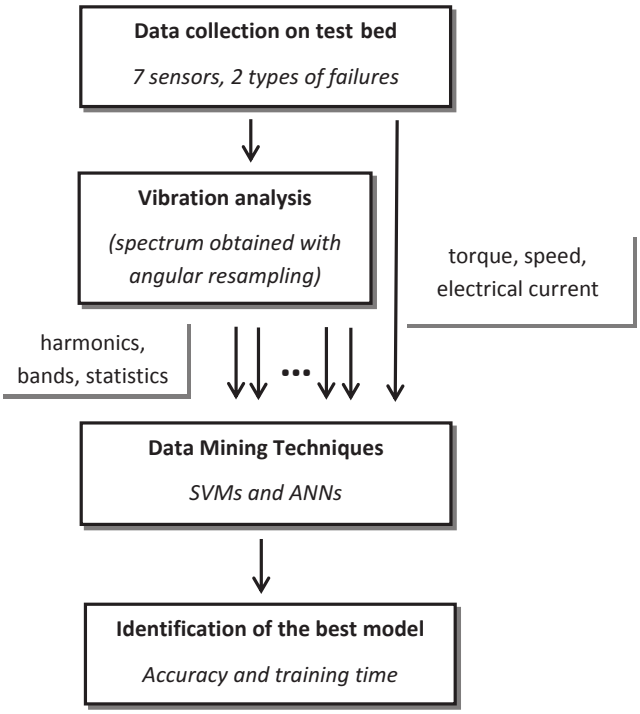


Figure 1. Diagram of the proposed methodology.

The paper is organized as follows. Section 2 introduces the experimental procedure for data collection, including a description of the test-bed, the sensors positioned on it and the failures that are generated, in order to simulate the real operating conditions of a wind turbine. Section 3 presents the analysis of the vibration signals obtained from the accelerometers and the final variables defined to generate the dataset. Section 4 presents the data-mining techniques used on the dataset for the classification task: SVMs and ANNs. Section 5 presents a detailed discussion of the results for these two methods in terms of accuracy, training and tuning times. Finally, Section 6 sets out the conclusions and future lines of work.

2. Experimental Procedure and Data Collection

The experiments presented in this paper have been designed to simulate the behavior of wind turbines (variable speed and load) on a test-bed, in which defective operation can be simulated in a controlled way. The test-bed is composed of two sub-assemblies (Figure 2) that are separated by a mounting flange specially designed to maintain the alignment of the subassemblies. The first sub-assembly on the right of the picture includes a quick speed shaft, with an electrical drive (rather than a generator), a parallel gearbox and a planetary gearbox with a slow speed shaft. This set of gears (parallel + planetary) represents the configuration and the gear ratio of a commercial wind turbine of around 1:61. The left sub-assembly, composed by a brake and a two-stage planetary gearbox, simulates the wind conditions with a random variation of rotating speed and load. These two parts are connected through a specially designed steel structure that can control alignment and misalignment on both sides, thereby generating one of the failure mechanisms. A complete list of the mechanical characteristics of the test-bed has been presented in a previous work [33].

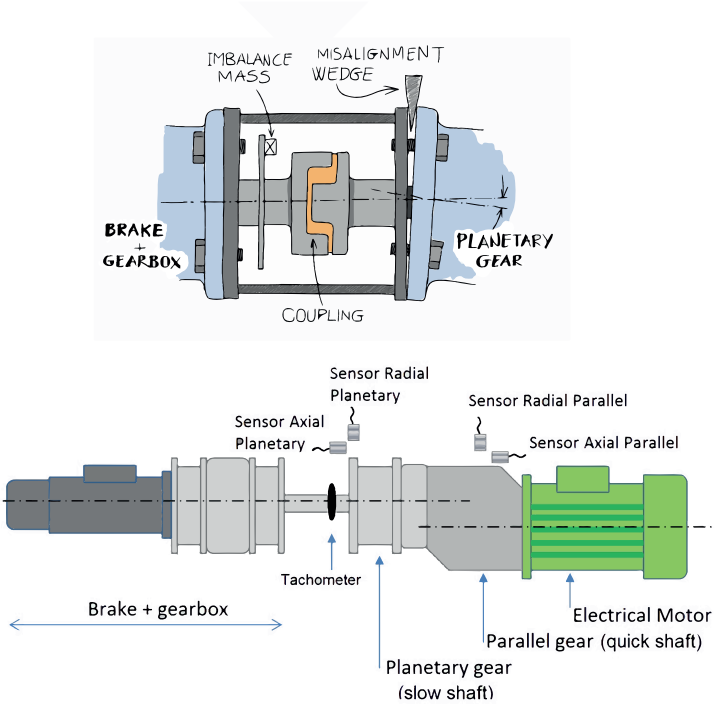


Figure 2. Scheme of the test-bed.

The vibrations were collected by four ICPaccelerometers in axial and radial directions from both gearboxes (parallel and planetary). The current and the torque of the electrical drive were also measured, and the speed of the slow shaft was registered by means of a laser sensor on a disk, using the same configuration that can be found in a real wind turbine for the measurement of low shaft speeds. In total, a 7-sensor system was developed for on-line monitoring of the test-bed performance. Figure 2 shows the positioning of all of the sensors.

The failures simulated on the test-bed were imbalance and misalignment, starting with small levels and increasing with each new set of measurements to simulate a progressive failure. Imbalance failure was simulated by attaching different masses to a disk used for speed measurement (for the imbalance), whereas misalignment failure was simulated by inserting a metallic wedge between the steel structure and the planetary gear on the right side of the test-bed (Figure 2). Altogether, seven failure cases and one non-failure case were tested. Table 1 shows the imbalance weight for each imbalance failure case, together with the equivalent percentage of the total weight of the rotor test-bed and the angle of each misalignment failure. The two levels of misalignment and the four levels of imbalance are enough to describe a progressive degradation of the mechanical chain of a wind turbine. Only one combination of both failures was tested, because this failure typology is very uncommon in real wind farms.

Table 1. Type and level of faults.

Imbalance			**Misalignment**	
Label	**Weight (g)**	%	**Label**	**Angle (°)**
Imbalance 1	5.79	0.077	Misalignment 1	0.78
Imbalance 2	9.13	0.12	Misalignment 2	1.53
Imbalance 3	19.5	0.26		
Imbalance 4	28.8	0.38		

Random profiles of speed and load were programmed on the test-bed to cover an extensive range of real working conditions (e.g., a certain failure, once it appears, will be working under the load and speed conditions defined by the wind that is blowing in real time). Every 100 s, load and speed were stepped up to the next random condition, to generate a random profile of working conditions. The start of the speed test interval was chosen as 1000 rpm, because this is approximately the equivalent speed at which the slow shaft of a wind turbine begins to generate energy. The selected speed range was between 1000 and 1800 rpm, and the load range was between 0% and 100%. An example of these profiles of load and speed is shown in Figure 3, where the Y-axis shows the programmed amplitude of the speed and load for each experiment and the X-axis shows the time domain with a new step in load and speed conditions every 100 s. On average, each working condition was programmed on the test-bed and acquired 100 times by the multi-sensory system, in the whole measurement stage.

The measurement time to capture the vibration signal for each of the four accelerometers was fixed at 72 s (this gives an equivalent frequency precision of 0.014 Hz) with a sampling frequency of 25,600 Hz. The rotating speed was also measured on the slow shaft (Figure 2); this signal is used for angular resampling of the vibration signals, in line with the technique presented in an earlier work [34]. Each of these 72-s measurements generated an instance of the final dataset, following the procedure that is

described in Section 3. In total, 6551 different working conditions were registered (considering failure level, load and speed), and so, the dataset will be composed of 6551 different instances.

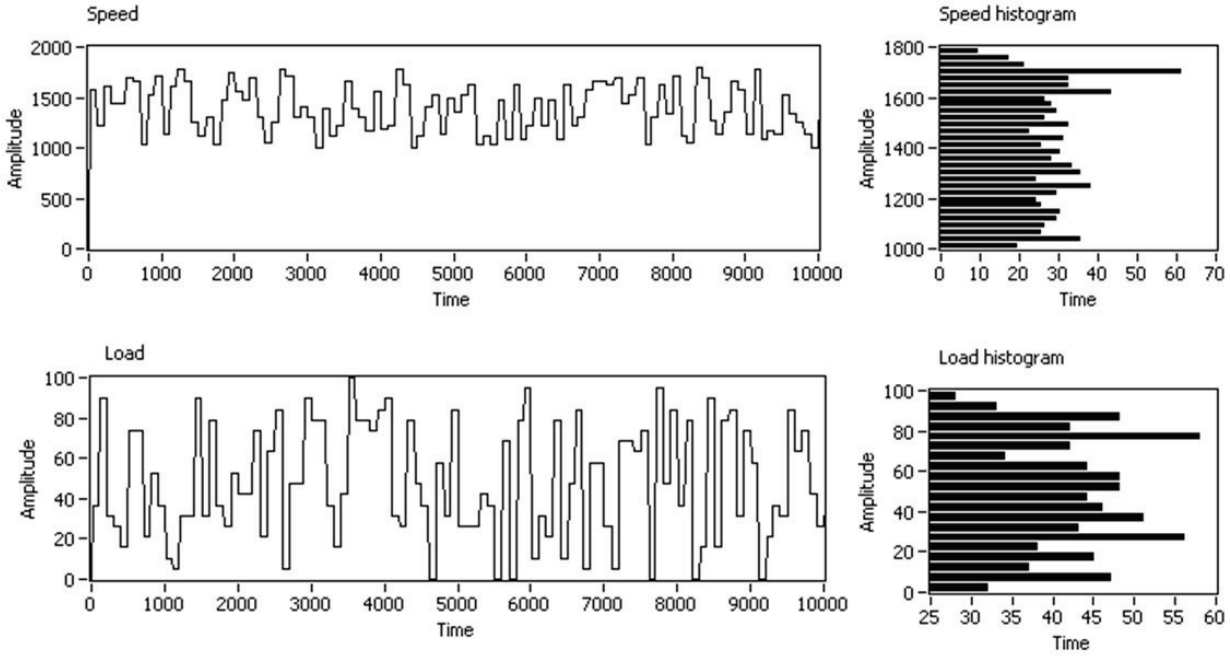


Figure 3. Example of random profiles of speed and load applied to the test-bed.

3. Vibration Analysis and Variables Definition

Having collected the multi-sensory information from the tests, it was necessary to filter out the variations in speed and load included in each vibration measurement. This variation was artificially created during the tests to simulate real conditions in wind turbines: wind speed and force is not stable and fluctuates over time. However, the information that vibration sensors could provide, if this variation were presented, is very limited. The angular resampling technique makes it possible to eliminate speed variability.

Once this technique had been applied to the raw data of the accelerometer signals, in line with the procedure described in a previous work [34], different variables were calculated for each measurement interval of 72 s. These variables are mainly obtained from the analysis of the vibration spectrum recorded by the accelerometers. Some variables take account of the energy distribution; others take account of the energy allocated in the harmonics of the rotating speed and in standard frequency bands (in this case, the frequency spectrum is split into bands at a fixed position unrelated to the natural frequency of the system). It should be remarked that such a methodology is independent of the critical frequencies of the mechanical components of the test bed, and therefore, the system needs no expert knowledge for failure detection. The calculated variables, summarized in Table 2, are:

- Global variables associated with the operational state of the test-bed: torque, speed, electric input current (3 variables) and electric output current (4 variables).

- Energy contained in certain frequency bands associated with the potential mechanical faults that can occur in the bearings and the gears of the test-bed for each vibration signal (272 variables for all four accelerometer).
- Energy contained in certain order bands associated with possible mechanical faults in the bearings and the gears of the test-bed for each vibration signal (245 variables for all four accelerometers).
- Global statistical variables for each vibration signal: average, RMS, skewness, kurtosis and interquartile range (20 variables, 5 variables for each accelerometer).

Table 2 summarizes the final number of variables, which is 544, and the variation range of all of the variables.

Table 2. Variables included in the final dataset.

Operation State			
Magnitude	**Number of Variables**	**Units**	**Range**
Torque	1	% of maximum torque	1–100
Speed	1	rpm	1000–1800
Input current	1	Amperes	1.63–2.85
Electrical current in the axis	4	Amperes	2×10^{-4}–0.12
Vibration Analysis			
Magnitude	**Number of Variables**	**Units**	**Range**
Harmonics	272	$10^{-3}\times$ mm/s^2	5.63×10^{-3}–0.075
Bands	245	$10^{-3}\times$ mm/s^2	1.65×10^{-2}–0.052
Average	4	$10^{-3}\times$ mm/s^2	−2–4
RMS	4	mm/s^2	0.016–0.12
Skewness	4	dimensionless	1.40–4.78
Kurtosis	4	dimensionless	2.19–66.1
Interquartile range	4	mm/s^2	0.021–0.17

The selected frequency bands and order bands have been presented in detail in a previous paper [33]. Some of the bands represent the characteristic frequencies of the different mechanical faults that may arise in machinery, such as imbalance, misalignment and gear faults. The calculation of these frequencies is well known in the literature and depends in many cases on geometrical parameters, such as the number of teeth of each gear (in our case, different gear ratios can be tested to define these bands, if the number of teeth is unknown). Other selected frequencies are the natural frequencies of the structure. These frequencies were selected in run-up test experiments, in which the rotation speed was increased with a ramp profile. After the experiments, the vibration was analyzed in the time-frequency domain, and the natural frequencies (constant) were separated from those mechanical frequencies that correlated with the speed (variable). This procedure may also be found in the literature [33].

As has been explained in Section 2, 6551 different conditions were tested. The distribution of the instances among the fault cases is summarized in Table 3. From this table, it can clearly be concluded

that the dataset was evenly balanced between the seven fault cases and the non-fault case. This balanced distribution of the instances is not common in such an industrial problem, because usually, the instances that describe failures and, specially, severe failures are very rare. The use of a test-bed in this research work is therefore clearly advantageous. It may be verified from Figure 4 that the random profile program generated an extensive map of different conditions. This figure shows the distribution of instances between the different speed (rpm of the slow shaft) and load (% of torque). Although the 6551 instances do not offer a completely homogeneous profile of the working conditions, all of the ranges are presented in the dataset with a significant number of instances.

Table 3. Distribution of measurements between the different instances of failures that were tested.

Type of Fault	Number of Instances
No misalignment, No imbalance (NO)	887 (13.54%)
No misalignment, Imbalance 1 (IM1)	847 (12.93%)
No misalignment, Imbalance 2 (IM2)	856 (13.07%)
No misalignment, Imbalance 3 (IM3)	838 (12.79%)
No misalignment, Imbalance 4 (IM4)	864 (13.19%)
Misalignment 1, No imbalance (MI1)	872 (13.31%)
Misalignment 2 No imbalance (MI2)	835 (12.75%)
Misalignment 2, Imbalance 4 (IM4 + MI2)	552 (8.43%)

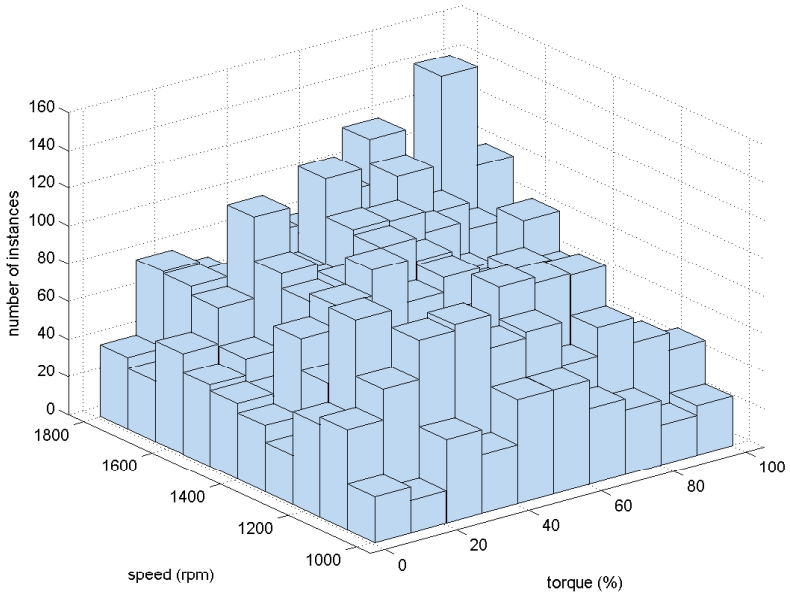


Figure 4. Histogram of the pair (torque, speed).

4. Data-Mining Techniques

Once the dataset is generated, different data-mining techniques could be applied to it, searching for the extraction of as much information as possible about the behavior of the test bed under failure conditions.

This section presents the data mining techniques applied to the dataset. First, an explanation is given of the two classification techniques that are tested; then, a discussion on feature-selection techniques is included, due to the huge amount of variables presented in the dataset. The two classification techniques used are ANNs, because they are the reference method for modeling this type of industrial problem [13], and SVMs, because of their characteristics that can outperform ANNs in this sort of industrial task.

4.1. Neural Networks

This classifier method is one of the most popular data-mining techniques. As indicated in the Introduction, it is probably the most frequently used in the context of fault diagnosis in rotating k-machines [13]. The extended use of this classifier could be due to two main reasons: first, it is a universal approximator of functions [35]; and, second, its mathematical formulation is inspired by biological functions, as it aims to emulate the behavior of a set of neurons. To do so, groups of small units, called neurons, are connected according to different architectures, the most habitual being the multi-layer perceptron (MLP). In the context of classification, this network has three layers, one with the network inputs (features of the dataset), a hidden layer and an output layer where the class is assigned to each input instance. The number of neurons in the input and output layer corresponds to the dimensions of the feature vector and the number of classes, respectively, but the number of neurons in the hidden layer is an important parameter of the structure of the ANN that needs to be adjusted.

When an instance is classified, firstly the output of the hidden layer, y_{hide}, is calculated from the inputs, and then, the output, y_{hide}, is taken as the input, according to the expressions shown in the formula [36]:

$$\begin{aligned} y_{hide} &= f_{net}(W_1 x + B_1) \\ y_{output} &= f_{output}(W_2 y_{hide} + B_2) \end{aligned} \tag{1}$$

where W_1 is the weight matrix of the hidden layer, B_1 is the bias of the hidden layer, W_2 is the weight matrix of the output layer (e.g., the identity matrix in the configuration tested), B_2 is the bias of the output layer, f_{net} is the activation function of the hidden layer and f_{output} is the activation function of the output layer. These two functions depend on the chosen structure, but are typically the identity for the hidden layer and $tansig$ for the output layer [36]. It is possible to include the bias in the weight matrix, by considering an additional fixed input $x_0 = 1$ and an output $y_0 = 1$. With this change and substituting the expression of y_{hide} into y_{output}, it is reformulated as follows:

$$y_{output} = f_{output}(W_1 \times [f_{net}(W_2 x_{extended})]) \tag{2}$$

The back-propagation algorithm is used to obtain the weights W_1 and W_2 in these ANNs. This algorithm arrives at an initial estimate of the weights and then performs an iterative process that updates the weights according to the error between the predicted output and the real output. There are two important parameters in the learning process: the learning rate, α, and the momentum, β. Calling t the iteration number and E the error function, for each individual element of the matrix W_1, w_{ij}, the updating rule may be outlined as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha \times \frac{\partial E}{\partial w_{ij}} + \beta \times (w_{ij}(t) - w_{ij}(t-1)) \tag{3}$$

Besides these two, a further parameter known as "epochs" can be used to limit the training time, stopping the loop for learning after a set number of iterations.

4.2. Support Vector Machines

The SVM technique [37] searches boundaries with the maximum margin of separation from the training data mapped into a space (called a space kernel) obtained from a transformation function. Margin maximization usually reduces the generalization error (*i.e.*, the expected error on a test set independent from the dataset used to build the classifier). We start the explanation of this classification technique with a geometrical view of the method, followed by its mathematical formulation. Firstly, the specific case of no data transformation is detailed (the case in which the kernel, as explained in subsequent sections, is linear, where the space kernel is equivalent to the original space) and, then, the general mathematical formulation and the types of kernels in use.

Beginning with the ideal case of a linearly-separable dataset, a simple geometrical view of the idea behind the SVM, without mapping the data onto another space, is possible, as is the concept of the margin. In the left part of Figure 5, we have a class with its instances in crosses and another class marked in circles. The three lines that are drawn would be a valid way of separating the training dataset, but the continuous one appears to be a more appropriate way. Mathematically, if we define the margin as the distance between the boundary and the nearest parallel line that crosses an instance of any of the classes, the boundary that is drawn represents the boundary with the maximum margin (see the right part of Figure 5).

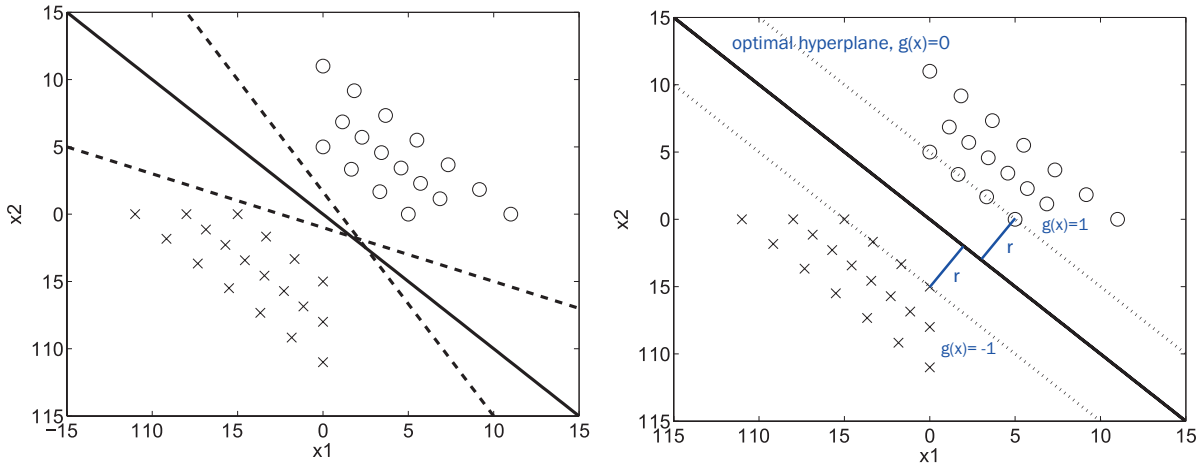


Figure 5. Idea of the SVM in the linear case.

In practice, there are a lot of problems that mean the data may not be linearly separated, which is why a search is made for boundaries with a more complex geometry. In fact, in the case of SVM, a transformed space of the data is used, in which a search for the hyperplane is performed, as we can see in Figure 6.

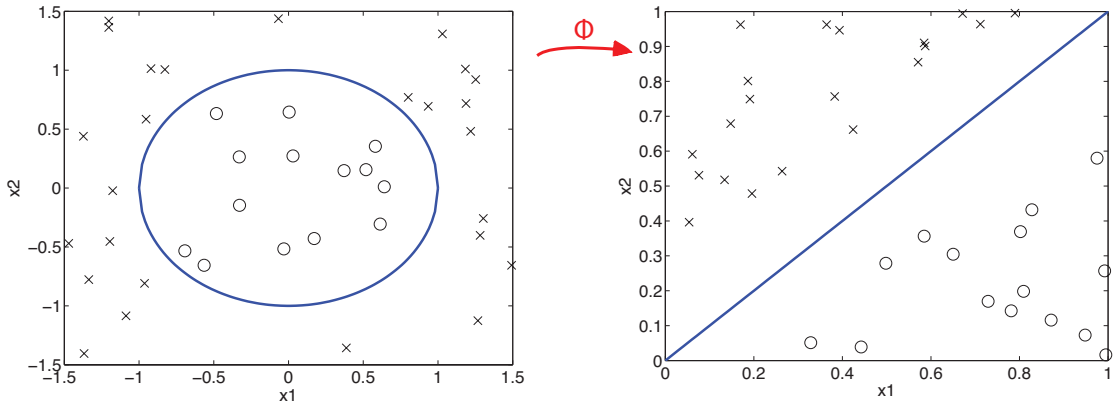


Figure 6. Graphical view of the SVM.

In this transformation, a function (ϕ) is applied that searches the projection for linearly-separable areas. This can produce a space with more dimensions than the input space and even an infinite number (a case of infinite kernel techniques).

We start the mathematical formulation with the linear case, given a training set $\{(x_i, y_i)\}_{i=1}^{N}$, which contains *D*-dimensional input vectors $x \in \Re^D$ and their corresponding labels $y \in \{-1, +1\}$. The hyperplane that we want to find by maximizing the margin between classes is expressed as $g(x) = w^T x + b$. The optimization problem can be written in terms of two parallel hyperplanes ($g(x) = 1$ and $g(x) = -1$, as shown in Figure 5):

$$\begin{aligned} &g(x) >= 1, \text{if } y = 1 \\ &g(x) <= -1, \text{if } y = -1 \end{aligned} \tag{4}$$

The margin of separation between the parallel hyperplanes is $2/||w||$. Maximizing the margin is equivalent to minimizing the inverse expression, $1/2||w||$. The formulation of the optimization problem is shown in Equation (5), where $||w||^2$ is used instead of $||w||$ for the convenience of carrying out the calculation process. The conditions $g(x) >= 1, if y = 1$ and $g(x) >= -1, if y = -1$ can be expressed in one single equation, by taking the product $g(x)y$. This grouping gives us the formulation of the linear case, as follows:

$$\begin{aligned} &\min_{w,b} y\frac{1}{2}||w||^2 \\ &s.t.\ \ y_i(w^T x_i + b) >= 1, \ \ i = 1, ..., n \end{aligned} \tag{5}$$

The previous scenario matches an idealized situation, and normally, the type of problem that is considered is non-linearly separable; a case in which the optimization problem that is formulated would have no solution. Therefore, the previous problem is generalized by means of a slack variable, ϵ, and a regularization parameter, C. The general formulation for the linear kernel is in this case:

$$\begin{aligned} &\min_{w,b} y\frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \epsilon_i \\ &s.t.\ \ y_i(w^T x_i + b) >= 1 - \epsilon_i, \epsilon_i > 0, \ \ i = 1, ..., n \end{aligned} \tag{6}$$

where ϵ is related to the degree of error that is permitted, while the term C controls the importance that is attributed to the second term included in the minimization objective, to allow an error margin (the null parameter would imply no error).

Having seen the formulation for the linear case, the general SVM formulation will be presented. In general, the hyperplanes used as a decision boundary can not be formulated directly in the original feature space. Instead, a non-linear transformation of the data is completed beforehand, by using the "kernel trick". The final expressions of the SVM optimization problem are detailed here. The hyperplanes are given by the the inner product (denoted by <.,.>) using a function ϕ that leads to the kernel transformation:

$$\begin{array}{l} \min_{w,b,\epsilon} y\frac{1}{2} < w, w >^2 + C\sum_{i=1}^{n} \epsilon_i \\ s.t.\ \ y_i(< w, \phi_x > +b) >= 1 - \epsilon_i, \epsilon_i > 0, \ \ i = 1, ..., n \end{array} \tag{7}$$

In this case, the classifier is built as follows:

$$g(x) = sign(< w, \phi_x > +b) \tag{8}$$

As the space in which the inner products are calculated can have infinite dimensions, the dual form of the problem is used by the SVM in its solution:

$$\begin{array}{l} \min_{\lambda} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N} \lambda_i \\ s.t.\ \ \sum_{i=1}^{N} y_i \lambda_i = 0,\ 0 <= \lambda_i <= C \end{array} \tag{9}$$

where K is the kernel function, defined for two different feature vectors x_i and x_j as $K(x_i, x_j) =< \phi_{x_i}, \phi_{x_j} >$, and the new optimizing objective is written in terms of the Lagrange multipliers, λ. Thus, the kernel function allows us to calculate the inner product between the mapped vectors without explicitly computing the mapping.

As an example of a kernel, we propose the radial basis kernel (or Gaussian kernel), given by a Gaussian distribution, with standard deviation parameter σ:

$$K(x_i, x_j) = exp(-\frac{||x_i - x_j||^2}{2\sigma^2}) \tag{10}$$

This type of kernel has two parameters that need to be optimized, the regularization parameter C and σ.

In this work, we have used four kernels, on the one hand, the two most commonly used—linear and Gaussian—and, on the other hand, two techniques with more novel kernels—stump and perceptron. These two kernels for infinite ensemble learning that are proposed in [38] were inspired by ensemble classifiers, combining good experimental results reached with both SVM and ensemble learning techniques. The aforementioned article explains how the kernel of an SVM should be generated, so that it is equivalent to an ensemble of infinite size. An ensemble is a combination of classifiers, usually trained with the same algorithm, but using different variations from the original dataset (e.g., subsampling, projections, instances reweighting, *etc.*) [39]. The ensemble prediction is usually computed by some weighted or non-weighted voting schema of the prediction of its members. Ensembles can improve their predictions by growing the number of their classifier members, so that is the aim of having a classifier, which theoretically emulates an ensemble of infinite size. Stump kernel is equivalent to an ensemble of infinite decision stumps. A decision stump is a decision tree with only one node. Therefore, it is a very simple classifier, as it makes its predictions by comparing only one feature of the instance against a precomputed threshold. Given that x_i and x_j are two instances, the kernel function $K(x_i, x_j)$ for the

stump kernel can be computed by summing the absolute values of the differences $||x_i - x_j||$ for all of the dimensions of the original data (*i.e.*, $l1$ norm). The perceptron kernel is equivalent to an ensemble of infinite perceptrons. A perceptron is perhaps the simplest ANN. It classifies instances by using a hyperplane in the original problem space as a decision boundary. $K(x_i, x_j)$ for the perceptron kernel can also be easily computed by the Euclidean distance between both instances (*i.e.*, $l2$ norm). Note that both the stump and the perceptron kernels have only one parameter to tune C, which is the common parameter for any SVM.

Finally, in [38], similar accuracy obtained by these kernels and the Gaussian kernel for real non-linearly separable cases is shown, but with less computational time, as it is necessary to adjust only one parameter, instead of two in the Gaussian case.

4.3. Variable Selection

The dataset includes 544 variables and 6551 instances. Such a large number of variables in comparison with the number of instances is due to the absence of an expert to select the most reliable variables to diagnosis failures in wind turbines. It is the only way to assure that the proposed intelligent system could be suitable for different mechanical chains without needing expert help for its configuration. Even though, from the mechanical and vibrational point of view, the selected set of variables that have previously been discussed are suitable for diagnosis misalignment and imbalance [33], this paper also presents a new analysis to confirm this conclusion, this time from a data-mining point of view. We used two variable selection techniques, in order to ensure that it was not possible to obtain classifiers with the same accuracy using fewer variables.

The variable or feature selection techniques are composed of two elements: a variable subset search method and a system of evaluation of their accuracy [40]. The second element, the accuracy evaluation system, allows us to group the feature selection techniques into two families [41]. The first one includes all techniques that use a filter variable selection technique, when the classification technique that will be used after the variable selection is not taken into account, but the attributes are chosen by means of statistical techniques that analyze the correlation that the attributes have between themselves and the class to be classified. The second group includes all techniques that use the classifier to evaluate the accuracy; in this case, there is a wrapper variable selection technique.

For this research, we have used one feature selection technique of each group. In the case of "filter type", one of the most common variable selection techniques was used: "correlation-based feature subset selection". In this method, the preference is for subsets of features that are highly correlated with the class, while having low intercorrelations. For the second group, an SVM classifier was used as an evaluator for the "wrapper" selection.

5. Results and Discussion

This section presents the results of variable selection and the performance of ANNs and SVMs in the dataset under study. The experiments were completed with WEKA [42] with an Intel Xeon E5405 (2 GHz) processor. Cross-validation was selected for both the training and the validation process, to assure that the results are independent of the instances selected for the training phase.

The ANN implementation of WEKAthat was used in the experimentation has three main parameters: one related to the structure of the classifier, the number of neurons of the hidden layer, and two related to the training phase, the momentum and the learning rate. Tuning of these three parameters of ANNs is a hard task if the dataset presents a large number of instances. An expert is needed to reduce this tuning time. For example, an automatic process to tune the ANN configuration of the three parameters was run on the dataset, without achieving an optimized result within a reasonable period of time. Specifically, using the WEKA "CVParameterSelection"tuning parameters method to adjust the three parameters at the same time, no repetition of a 5 $\times$ 2 cross-validation could be completed after five days of execution with the Intel Xeon E5405 processor. For this reason, the optimization process was run on only two training parameters (momentum and learning rate) for each specific configuration of the structure of the ANN (number of neurons in the hidden layer). In this way, the number of configurations tested in the same repetition was drastically reduced, and the computer was able to finish the tuning process in a reasonable time. The optimization method used a grid search provided by WEKA, with a 5 $\times$ 2 cross-validation, for the tuning of the pair momentum-learning rate. This method has two steps; in the first step, a two-fold cross-validation is used to estimate an initial optimum pair of the grid; then, an iterative process is performed, to confirm that the combination found in the first step is the best one. To do so, the best combination found in the first step is used as a center point and compared with its adjacent points through a 10-fold cross-validation, changing the pair and repeating the process if a better solution is found. Following this method, the first tests with 0, 5, 10, 15, 20 and 30 neurons in the hidden layer were performed. Table 4 shows the accuracy of the best configuration of ANNs found in each case, expressed in percentages of correctly classified instances. This analysis shows that the peak of maximum accuracy was around 20 neurons, in bold in Table 4, and any increase over this number of neurons does not improve the accuracy.

Table 4. Accuracy of the ANN changing neurons in the hidden layer.

Hidden Layers	0	5	10	15	**20**	30
Accuracy	95.79	96.66	97.15	97.20	**97.34**	96.96

Then, configurations of around 20 neurons (*i.e.*, N = 15, 16,... , 23) were tested, and the optimization grid was performed to find the best momentum and learning rate for each configuration. Table 5 shows the result of this execution: the best result, in bold, was obtained for 17 neurons in the hidden layer (97.47%). We therefore have to consider that the most suitable structure of the ANN for this dataset is a 17-neuron configuration.

Table 5. Accuracy for a number of neurons of around 20.

Hidden Layers	15	16	**17**	18	19	20	21	22	23
Accuracy	97.20	97.12	**97.47**	97.05	97.26	97.34	96.96	97.39	97.15

After the identification of the best ANN for this dataset, the SVMs were optimized. Sequential minimal optimization [43] is the SVM implementation used by WEKA. The stump and perceptron

kernel had to be implemented within WEKA. SVM classifiers with the four types of kernels described in Section 4 were tested over the dataset, with a 5×2 cross-validation, while optimizing the corresponding parameters for each repetition and fold. In the Gaussian case, the regularization parameter, as well as the gamma parameter were optimized, while in the other kernels, it was only necessary to consider the regularization parameter. The same WEKA grid method used for optimizing momentum and learning rate of ANNs was used to optimize the SVM with a Gaussian kernel, whereas the other SVMs were optimized using a single parameter selection method.

Table 6 summarizes the results obtained from the optimization of SVM on the dataset and, for comparison, the accuracy of the best ANN configuration. Table 6 presents not only the accuracy of each data-mining technique, but also the tuning and training times, because both parameters are critical in real cases where an expert is not available to speed up the optimization process or where the computer is programmed to do so in an industrial environment. We used the corrected resampled *t*-test provided by WEKA [44] with a 95% confidence level to compare the methods. The best method, in terms of all three variables (accuracy, tuning and training time) is the linear SVM. The cases that are significantly worse than this best method are indicated with an asterisk. The accuracy obtained by an SVM with linear kernels is also of a higher statistical significance than the results of a previous study of this dataset using ensembles (96.24%) [20].

Table 6. Summary results of SVMs *vs*. ANNs.

	Linear SVM	**Perceptron SVM**	**Gaussian SVM**	**Stump SVM**	**ANNs**
Accuracy (%)	**98.26** (0.24)	96.86 (0.36) *	97.25 (0.31) *	**97.85** (0.23)	**97.47** (0.24)
Tunning time (s)	444.27 (10.38)	1,241.27 (16.27) *	16,068.97 (35.42) *	945.17 (11.19) *	46,611.12 (2,391.96) *
Training time (s)	**12.48** (0.40)	21.55 (0.31) *	15.61 (0.28) *	22.17 (0.28) *	491.00 (10.77) *

* The method is significantly better

Besides the overall accuracy, a detailed decomposition of the error between the classes is shown, by means of the confusion matrix for the linear SVM case (Table 7). That matrix highlights the following issues:

1. There was 0% error in discriminating imbalanced failures *vs*. misalignment failures and discriminating any of these failures against the combination of both of them.
2. Only rarely (0.02%) did the classifier identify no failures, but Imbalance 2. Furthermore, on very few occasions (0.02%), imbalances of two and four were not recognized by the system.
3. Almost all errors in the confusion matrix occurred in the categorization of imbalance levels. Specifically, most of the matrix errors appeared in the discrimination between Imbalance 3 *vs*. Imbalance 4 (*i.e.*, 0.48% and 0.76%, which constitute 90% of all the matrix errors). This fact can be explained by the weights used to simulate each imbalance failure. As shown in Table 1, Imbalance 2 was simulated by applying approximately twice the weight used in Imbalance 1, and

Imbalance 3 was also simulated by approximately twice the weight of Imbalance 2. However, this criteria of 100% weight increases was no longer valid when simulating Imbalance 4, where the weight only increased by around 50% in relation to Imbalance 3.

Table 7. Confusion matrix for the linear SVM classifier.

		Forecasted Class							
		NO	**IM1**	**IM2**	**IM3**	**IM4**	**MI1**	**MI2**	**IM4 + MI2**
Real class	**NO**	13.52	0	0.02	0	0	0	0	0
	IM1	0.01	12.73	0	0.16	0.03	0	0	0
	IM2	0	0	13.02	0.04	0	0	0	0
	IM3	0	0.14	0.04	11.85	0.76	0	0	0
	IM4	0.01	0.04	0	0.48	12.66	0	0	0
	MI1	0	0	0	0	0	13.31	0	0
	MI2	0	0	0	0	0	0	12.75	0
	IM4+MI2	0	0	0	0	0	0	0	8.43

Taking account of accuracy alone, the best ANN configuration performed in a similar way to both SVMs with linear kernels and SVMs with stump kernels (see Table 6). However, when training and tuning time is also considered, SVMs with linear kernels clearly appear to be the most suitable method for this dataset, with a performance of half of the tuning time of other SVMs and at least 25% better in training time. If we compare ANN *versus* SVM performance, we can conclude that all SVMs require significantly less training time (at least 20-times less) than an ANN. In relation to the tuning time, the linear SVM is 10-times quicker than the best ANN configuration. The big difference between training and tuning times is remarkable and could be reasonably explained. Although ANNs and SVM with RBFkernels need to adjust two parameters, SVMs using the other kernels (linear, stump and perceptron) only need to fit one parameter, and therefore, in large datasets, there is a clear difference in tuning time. In fact, adjusting the parameters of an ANN demands more time than is actually shown in Table 6, which shows the time once the optimal number of hidden layers is known.

Linear SVM is the only method from the experiments that provides hyperplanes in the dataset space as decision boundaries. All of the remaining tested methods compute more complex decision boundaries (*i.e.*, non-linear boundaries). The better accuracy of linear SVM therefore suggests that the dataset is linearly separable. The dataset was divided into 28 new datasets to test this hypothesis, each one containing only all of the instances from two of the eight classes (the problem has eight classes, so there are 28 possible combinations of two classes). For each one of these subsets, a linear SVM is computed using all of its instances. These SVMs are validated using the same data that were used for training. If 0% error is not reached for a dataset, the C parameter of the SVM is then increased, and the linear SVM is computed again. Increasing C narrows the margin by trying to include more instances (*i.e.*, those close to the border) in the right side of the hyperplane. Remember that C penalizes the error from misclassified instances in the optimization problem (see Equation (6)). The results of this experiment show that even in the most difficult binary problem (Imbalance 3 *vs*. Imbalance 4), it is always possible

to find a C value leading to a hyperplane with 0% training error. We may therefore conclude that the dataset is linearly separable.

Finally, a feature selection analysis was performed to assure the suitability of the selected variables, in order to diagnosis the two kind of failures. Table 8 summarizes the results of this analysis. Accuracy is expressed by the percentage of correctly classified instances. The standard deviation is included between parentheses for each magnitude. As previously explained in Section 4, two feature selection techniques were tested: a filter selection and a wrapper selection. The accuracy of both techniques is never higher than 95.6%; but a linear SVM with the completed dataset is able to achieve a significantly higher accuracy of 98.0%, without any tuning of its parameter. It may be concluded that the original dataset cannot be reduced in terms of variables without losing information.

Table 8. Accuracy of a SVM classifier with variable selection.

Filter Selection	Wrapper Selection	Default SVM
95.57 (0.38)	95.17 (0.37)	98.02 (0.26)

Feature selection results point out that all of the 544 features used by the hyperplanes that are computed with linear SVM are necessary to obtain significantly better results. Therefore, linear SVM performs better, because the dataset has been built using features that are very suitable for creating a linearly-separable dataset.

6. Conclusions and Futures Lines of Work

A promising architecture to detect bearing failures in wind-turbine gearboxes has been presented that comprises a combination of angular resampling for vibration analysis, monitoring of the wind-turbine power output and data-mining techniques for the classification task. To overcome the lack of datasets with a wide range of conditions of loads and speeds, different experiments have been performed on a test-bed for two common failure mechanisms: misalignment (two levels of failure) and imbalance (five levels of failure), in addition to the non-fault case. The whole dataset included 6551 instances with 544 variables, such that it can be considered a high dimensional problem. The dataset was mainly balanced within the seven fault cases and the non-fault case, the contribution of each one to the final data set being higher than 8.4% and lower than 13.5%.

SVMs are state-of-the-art classifiers that are more widely used in real problems because of their performance. In this paper, the most traditional kernels (*i.e.*, linear and Gaussian), besides some new kernel techniques that could be suitable for this industrial problem (*i.e.*, perceptron kernel and stump kernel) have been tested. The results have also been compared with ANNs, which are taken as a baseline method, as they are broadly used in other industrial classification problems. All parameters of the selected classifiers were tuned in the study. As the ANN tuning processes are time-consuming, instead of optimizing the number of the neurons'momentum learning rate in the output space, optimization of the momentum learning rate is performed for different values of the number of neurons.

The results showed that linear SVM performed best and is significantly better than other sophisticated kernels, such as the Gaussian kernel or the perceptron kernel. The stump kernel and ANNs were

outperformed by linear SVM, with only slight differences, but only if accuracy is considered. The confusion matrix for linear SVM shows 90% of errors in discrimination between Imbalance 3 *vs*. Imbalance 4, but the other binary problems either have no error or they are smaller than 0.02%. The concentration of errors in this binary problem is also explained by the way they were simulated. However, the performance of these techniques has also been compared in terms of tuning and training times, because both parameters are critical in real industrial applications when the computer data cannot be interpreted by an expert prior to the optimization task. It is the simplicity of the linear SVM method that makes it quicker, with training times that are much shorter than the time required by the other algorithms that have been tested. It outperforms the other algorithms when tuning time is considered, because of its short training time with only one parameter to tune. In contrast, ANNs have more parameters to tune, and their training times are very high; thus, it takes much longer for an ANN to achieve competitive results in its classification performance. All of these reasons make linear SVM the most suitable classifier for the problem proposed in this study. The reason behind the better accuracy of linear SVM has also been explored. The experiments point out that a linearly-separable dataset is obtained using the proposed methodology.

Future work will focus on the application of this methodology to other types of wind-turbine failures, so that the same software solution can fully identify the performance of the wind turbines. Moreover, the application of this methodology to real data from different wind farms is foreseen, where there will be a clear imbalance in the dataset between non-failure cases and failure cases.

Acknowledgments

This investigation has been partially supported by the Projects, CENIT-2008-1028, TIN2011-24046, IPT-2011-1265-020000 and DPI2009-06124-E/DPI of the Spanish Ministry of Economy and Competitiveness. Special thanks to Roberto Arnanz for data recording and processing throughout the project and Juan J. Rodriguez from the University of Burgos for such kind-spirited and useful advice.

Author Contributions

L.F. Villa, acting under the supervision, advice and guidance of A. Reñones, completed the design of the test-bed, its failure levels, the experimental activities and the angular resampling technique. J. Maudes was responsible for the identification of the different Artificial Intelligence Techniques to be tested on the final dataset, especially the different kernels to be tested with the SVMs, the tuning techniques for the ANNs and the final analysis of the results from the perspective of artificial intelligence. P. Santos performed the ANN and the SVM experiments under the supervision, advice and guidance of A. Bustillo and J. Maudes. Finally, A. Bustillo proposed the general approach in this paper as well as the final analysis of the results from an industrial point of view.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Sloth, C.; Esbensen, T.; Stoustrup, J. Robust and fault-tolerant linear parameter-varying control of wind turbines. *Mechatronics* **2011**, *21*, 645–659.
2. Joselin Herbert, G.; Iniyan, S.; Sreevalsan, E.; Rajapandian, S. A review of wind energy technologies. *Renew. Sustain. Energy Rev.* **2007**, *11*, 1117–1145.
3. Yang, W.; Tavner, P.J.; Crabtree, C.J.; Feng, Y.; Qiu, Y. Wind turbine condition monitoring: technical and commercial challenges. *Wind Energy* **2012**, *17*, 673–693.
4. Hameed, Z.; Hong, Y.; Cho, Y.; Ahn, S.; Song, C. Condition monitoring and fault detection of wind turbines and related algorithms: A review. *Renew. Sustain. Energy Rev.* **2009**, *13*, 1–39.
5. Tavner, P.; Xiang, J.; Spinato, F. Reliability analysis for wind turbines. *Wind Energy* **2007**, *10*, 1–18.
6. Davies, A. *Handbook of Condition Monitoring: Techniques and Methodology*; Chapman & Hall: London, UK, 1998.
7. Simani, S.; Fantuzzi, C. Dynamic system identification and model-based fault diagnosis of an industrial gas turbine prototype. *Mechatronics* **2006**, *16*, 341–363.
8. Feng, Z.; Liang, M.; Zhang, Y.; Hou, S. Fault diagnosis for wind turbine planetary gearboxes via demodulation analysis based on ensemble empirical mode decomposition and energy separation. *Renew. Energy* **2012**, *47*, 112–126.
9. Combet, F.; Zimroz, R. A new method for the estimation of the instantaneous speed relative fluctuation in a vibration signal based on the short time scale transform. *Mech. Syst. Signal Process.* **2009**, *23*, 1382–1397.
10. Jeffries, W.; Chambers, J.; Infield, D. Experience with bicoherence of electrical power for condition monitoring of wind turbine blades. *IEE Pro.-Vis. Image Sign.* **1998**, *145*, 141–148.
11. Caselitz, P.; Giebhardt, J.; Mevenkamp, M. On-line Fault Detection and Prediction in Wind Energy Converters. In Proceedings of the EWEC94, Thessaloniki, Greece, 10–14 October 1994; Volume 94, pp. 623–627.
12. Caselitz, P.; Giebhardt, J.; Kewitsch, R. Advanced Condition Monitoring System for Wind Energy Converters. In Proceedings of the EWEC99, Nice, France, 1–5 March 1999; pp. 63–66.
13. Samuel, P.D.; Pines, D.J. A review of vibration-based techniques for helicopter transmission diagnostics. *J. Sound Vib.* **2005**, *282*, 475–508.
14. Zhan, Y.; Makis, V.; Jardine, A.K. Adaptive state detection of gearboxes under varying load conditions based on parametric modelling. *Mech. Syst. Signal Process.* **2006**, *20*, 188–221.
15. Wenyi, L.; Zhenfeng, W.; Jiguang, H.; Guangfeng, W. Wind turbine fault diagnosis method based on diagonal spectrum and clustering binary tree SVM. *Renew. Energy* **2013**, *50*, 1–6.
16. Salahshoor, K.; Kordestani, M.; Khoshro, M. Fault detection and diagnosis of an industrial steam turbine using fusion of SVM (support vector machine) and ANFIS (adaptive neuro-fuzzy inference system) classifiers. *Energy* **2010**, *35*, 5472–5482.
17. Chen, J.; Hao, G. Research on the Fault Diagnosis of Wind Turbine Gearbox Based on Bayesian Networks. *Pract. Appl. Intell. Syst.* **2012**, *124*, 217–223.

18. Harris, T. A Kohonen SOM based, Machine Health Monitoring System Which Enables Diagnosis of Faults not Seen in the Training Set. In Proceedings of the 1993 International Joint Conference on Neural Networks, IJCNN'93, Nagoya, Japan, 25–29 October 1993; Volume 1, pp. 947–950.
19. Essawy, M. Fault Diagnosis of Helicopter Gearboxes Using Neuro-Fuzzy Techniques. In Proceedings of the 52nd Meeting of the Society for Machinery Failure Prevention Technology, Virginia Beach, VA, USA, 30 March–3 April 1998; pp. 293–302.
20. Santos, P.; Villa, L.; Reñones, A.; Bustillo, A.; Maudes, J. Wind turbines fault diagnosis using ensemble classifiers. *Adv. Data Min. Appl. Theor. Asp.* **2012**, *7377*, 67–76.
21. Garg, A.; Tai, K. An Ensemble Approach of Machine Learning in Evaluation of Mechanical Property of the Rapid Prototyping Fabricated Prototype. *Appl. Mech. Mater.* **2014**, *575*, 493–496.
22. Wang, G.; Yang, Y.; Zhang, Y.; Xie, Q. Vibration sensor based tool condition monitoring using ν support vector machine and locality preserving projection. *Sens. Actuators A Phys.* **2014**, *209*, 24–32.
23. Wang, G.; Yang, Y.; Li, Z. Force Sensor Based Tool Condition Monitoring Using a Heterogeneous Ensemble Learning Model. *Sensors* **2014**, *14*, 21588–21602.
24. Bustillo, A.; Díez-Pastor, J.F.; Quintana, G.; García-Osorio, C. Avoiding neural network fine tuning by using ensemble learning: application to ball-end milling operations. *Int. J. Adv. Manuf. Technol.* **2011**, *57*, 521–532.
25. Kim, H.; Pang, S.; Je, H.; Kim, D.; Yang Bang, S. Constructing support vector machine ensemble. *Pattern Recognit.* **2003**, *36*, 2757–2767.
26. Verikas, A.; Gelzinis, A.; Bacauskiene, M.; Olenina, I.; Olenin, S.; Vaiciukynas, E. Phase congruency-based detection of circular objects applied to analysis of phytoplankton images. *Pattern Recognit.* **2011**, *45*, 1659–1670.
27. Khan, N.; Ksantini, R.; Ahmad, I.; Boufama, B. A novel SVM + NDA model for classification with an application to face recognition. *Pattern Recognit.* **2011**, *45*, 66–79.
28. Li, M.; Han, K.; Narayanan, S. Automatic speaker age and gender recognition using acoustic and prosodic level information fusion. *Comput. Speech Lang.* **2012**, *27*, 151–167.
29. Hmeidi, I.; Hawashin, B.; El-Qawasmeh, E. Performance of KNN and SVM classifiers on full word Arabic articles. *Adv. Eng. Inform.* **2008**, *22*, 106–111.
30. Bustillo, A.; Correa, M.; Reñones, A. A Virtual Sensor for Online Fault Detection of Multitooth-Tools. *Sensors* **2011**, *11*, 2773–2795.
31. Bustillo, A.; Rodríguez, J.J. Online breakage detection of multitooth tools using classifier ensembles for imbalanced data. *Int. J. Syst. Sci.* **2014**, *45*, 2590–2602.
32. Bustillo, A.; Ukar, E.; Rodriguez, J.J.; Lamikiz, A. Modelling of process parameters in laser polishing of steel components using ensembles of regression trees. *Int. J. Comput. Integr. Manuf.* **2011**, *24*, 735–747.
33. Villa, L.F.; Reñones, A.; Perán, J.R.; de Miguel, L.J. Statistical fault diagnosis based on vibration analysis for gear test-bench under non-stationary conditions of speed and load. *Mech. Syst. Signal Process.* **2012**, *29*, 436–446.

34. Villa, L.F.; Reñones, A.; Perán, J.R.; de Miguel, L.J. Angular resampling for vibration analysis in wind turbines under non-linear speed fluctuation. *Mech. Syst. Signal Process.* **2011**, *25*, 2157–2168.
35. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366.
36. Delashmit, W.H.; Manry, M.T. Recent Developments in Multilayer Perceptron Neural Networks. In Proceedings of the Seventh Annual Memphis Area Engineering and Science Conference, Memphis, TN, USA, 11 May 2005.
37. Boser, B.; Guyon, I.; Vapnik, V. A Training Algorithm for Optimal Margin Classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992; pp. 144–152.
38. Lin, H.; Li, L. Novel Distance-Based SVM Kernels for Infinite Ensemble Learning. In Proceedings of the 12th International Conference on Neural Information Processing, Taipei, Taiwan, 30 October–2 November 2005; pp. 761–766.
39. Kuncheva, L. *Combining Pattern Classifiers: Methods and Algorithms*; Wiley-Interscience: Hoboken, NJ, USA, 2004.
40. Hall, M.A.; Holmes, G. Benchmarking attribute selection techniques for discrete class data mining. *IEEE T. Knowl. Data. Eng.* **2003**, *15*, 1437–1447.
41. Liu, H.; Setiono, R.; others. A Probabilistic Approach to Feature Selection–A Filter Solution. In Proceedings of the 10th Conference Machine Learning International Workshop, Bari, Italy, 3–6 July 1996.; pp. 319–327.
42. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I. The WEKA data mining software: An update. *ACM SIGKDD Explor. Newsl.* **2009**, *11*, 10–18.
43. Platt, J. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In *Advances in Kernel Methods*; MIT Press: Cambridge, MA, USA, 1999.
44. Nadeau, C.; Bengio, Y. Inference for the generalization error. *Mach. Learn.* **2003**, *52*, 239–281.

J Intell Manuf
DOI 10.1007/s10845-015-1110-0

Identifying maximum imbalance in datasets for fault diagnosis of gearboxes

Pedro Santos[1] · **Jesús Maudes**[1] · **Andres Bustillo**[1]

Received: 21 January 2015 / Accepted: 4 June 2015

Abstract Research into fault diagnosis in rotating machinery with a wide range of variable loads and speeds, such as the gearboxes of wind turbines, is of great industrial interest. Although appropriate sensors have been identified, an intelligent system that classifies machine states remains an open issue, due to a paucity of datasets with sufficient fault cases. Many of the proposed solutions have been tested on balanced datasets, containing roughly equal percentages of wind-turbine failure instances and instances of correct performance. In practice, however, it is not possible to obtain balanced datasets under real operating conditions. Our objective is to identify the most suitable classification technique that will depend least of all on the level of imbalance in the dataset. We start by analysing different metrics for the comparison of classification techniques on imbalanced datasets. Our results pointed to the Unweighted Macro Average of the F-measure, which we consider the most suitable metric for this diagnosis. Then, an extensive set of classification techniques was tested on datasets with varying levels of imbalance. Our conclusion is that a Rotation Forest ensemble of C4.4 decision trees, modifying the training phase of the classifier with a cost-sensitive approach, is the most suitable prediction model for this industrial task. It maintained its good performance even when the minority classes rate was as low as 6.5 %, while the majority of the other classifiers were more sensitive to the level of database imbalance and failed standard performance objectives, when the minority classes rate was lower than 10.5 %.

Keywords Fault diagnosis · Multi-class imbalance · Wind turbines · Ensembles · Metrics · Gearbox

Introduction

The rapid development of wind farms all over the world, since the start of the new Millennium has implied significant challenges, in the search to reduce their operational costs (Joselin Herbert et al. 2007). The operational costs of wind farms are mainly incurred by maintenance tasks, due to the difficulty of accessing wind turbines and the significant distance between wind farms and industrial areas. An automated monitoring, diagnostics and predictive system for wind turbines appears to be the best strategy to reduce their maintenance costs (Cao et al. 2014). However, wind turbines include many different components and the system diagnostics developed to date have habitually focused on only one sub-assembly: the powertrain, the power generator, the lubricating system, etc. In line with this strategy, this research focuses on the powertrain of the wind turbine.

The literature on wind farm maintenance (Hameed et al. 2009) outlines the critical role of the powertrain or the gearbox and the rotor blades of a wind turbine in its performance and the significant percentages of wind turbine failures next to the reasons for the breakdown. In these studies, the most common reasons for component failure are highlighted as rotor blade misalignment and powertrain failure due to bearing fatigue and gear damage. It is these last two causes of failure -powertrain failure due to bearing fatigue and gear damage- which will be considered in this research.

✉ Andres Bustillo
abustillo@ubu.es

Pedro Santos
psgonzalez@ubu.es

Jesús Maudes
jmaudes@ubu.esPedro

[1] University of Burgos, Burgos, Castilla y León, Spain

Published online: 15 June 2015

Wind turbines are usually equipped with different types of sensors for fault diagnosis in their gearboxes and powertrains. First, the power signals from the wind turbine are used, although they have been considered insufficient in themselves for this diagnosis. Only very few examples use spectral analysis of the power output signals to identify rotor blade imbalance (Jeffries et al. 1998) and gearbox and bearing faults (Caselitz et al. 1994). Second, accelerometers are used to register the vibrations in the rotation of the powertrain. However, the importance of vibratory analysis in the diagnosis of rotating machinery (Davies 1998) is still an open issue, especially in the case of wind turbines, because they present a new challenge in comparison with other rotatory machinery, functioning under variable speed and load conditions (Simani and Fantuzzi 2006). Wind turbines share this special characteristic with only a few other machines, such as excavators and helicopters (Villa et al. 2011). While in other rotatory machinery the Fast Fourier Transform (FFT) is usually used to monitor the condition of gearboxes, the information provided by this transformation is insufficient in the case of wind turbines. This limitation has led to the development of two different strategies to diagnose the failures in gearboxes of wind turbines (Nie and Wang 2013). On the one hand, there are some studies that propose to divide the sensors into subsets, obtaining separate information from each subset and then fusing these informations into the diagnosis system. Besides considering vibration signals collected by accelerometers, several studies have included acoustic emission signals from microphones (Lu et al. 2012; Lekou et al. 2009; Soua et al. 2013).

The decision system has to incorporate a procedure to fuse the information from the different subsets of sensors, using an aggregation technique such as Ordered Weighted Average (OWA) (Yager 2004). This procedure requires the calculation of the optimal weights associated with each base classifier, which are determined using a predictive error criteria (Filev and Yager 1994, 1998).

On the other hand, several authors have analysed how to develop advanced signal processing techniques for vibration signals that can be completed with data from other sensors that monitor, for example, rotation speed and the efficiency of the energy generation process (Hameed et al. 2009; Villa et al. 2011, 2012), which is the strategy followed in the present work.

Different approaches to data analysis of vibrational signals have been proposed. The statistical features of the vibrational signal in the time domain were considered in the initial attempts (Samuel and Pines 2005). Subsequently, spectral analysis and time-frequency analysis showed their efficiency at gathering more information than statistical analyses (Stander and Heyns 2006; Zhan et al. 2006; Bartelmus and Zimroz 2009). Recently, spectral analysis has been complemented by a technique called angular resampling, which extracts more information, to filter out the effect of variable wind speeds (Villa et al. 2011, 2012).

However, the acquisition and processing of signals from different sensors is not the last stage in the construction of wind turbine and rotating machinery system diagnostics. A decision-making system remains to be built, for which purpose, statistical models, such as the spectral kurtosis method (Barszcz and Randall 2009), the singular spectral analysis (Sánchez and Couso 2012), wavelet packet decomposition (Li et al. 2013), empirical mode decomposition with independent component analysis (Wang et al. 2014) and regression models (Zhan et al. 2006; Villa et al. 2011, 2012) have been applied. Moreover, techniques based on artificial intelligence (AI) seem to be very promising for this task, partly because no expert is needed to identify critical variables and to provide in-depth knowledge of the characteristics of the powertrain to build the AI models. Support Vector Machines (SVMs) (Ziani et al. 2014; Salahshoor et al. 2010), Bayesian networks (Chen and Hao 2012), Self-Organizing maps (Harris 1993), Artificial Neural Networks (ANNs) (Essawy 1998), Classifier Ensembles (Santos et al. 2012) and Neurofuzzy Inference Systems (Salahshoor et al. 2010) have all been tested with the same objective in mind: to prepare a decision-making system. ANNs are considered a standard technique for many industrial tasks, from traditional manufacturing optimization like drilling (Bustillo and Correa 2012) or turning (Gajate et al. 2012) to new techniques like laser milling (Teixidor et al. 2013). However, other types of classifiers have proved its suitability to model industrial problems (Garg and Tai 2014), but they have been rarely used in the context of rotating machine maintenance.

Most of these works present clear differences as regards the real working conditions of wind turbines, in so far as they use balanced datasets. The number of instances that represent a fixed failure is similar to the number of instances that represent the normal working condition of the wind turbine (Villa et al. 2011, 2012; Santos et al. 2012). There are two reasons for this fact: first, the datasets are mainly generated on test-beds where the failures are artificially provoked and the risk of damage to the test-bed while running it under failure conditions is assumed to generate turbine performance data that will mirror those same failure conditions. A real wind turbine running under failure conditions is immediately stopped as soon as the fault is detected, to avoid further damage, so there is no chance of gathering data after that point. The datasets recorded under real wind turbine working conditions will therefore be very imbalanced, due to a paucity of instances that represent failures. This situation is even worse in comparison with other rotary machinery, due to the variable load and speed of wind turbines: while conventional machine failure can be defined by a small number of instances, because its working conditions are very limited (e.g. a limited range of rotary speed at a fixed value of load), the complete identi-

fication of wind turbine failure would require the operation of the faulty wind turbine under an extensive range of wind conditions.

Although this observation in no way invalidates the conclusions of the aforementioned works, it does point to the need for a deep analysis of the effect of imbalance on the accuracy of AI models applied to wind turbine datasets; an analysis that constitutes the main objective of this research, seeking the practical implementation of its results in real wind farms. From an industrial point of view, this analysis might identify the maximum imbalance level that a certain AI model can tolerate, before excessive loss of quality occurs in its performance. In this way, the maintenance engineer tasked with the system diagnostics of the wind farm will be able to select the most appropriate AI technique for the diagnostics. This selection process will depend on the availability of data acquired under failure conditions, a parameter that is directly linked to the age of the wind farm and to its monitoring equipment.

However, a further question arises when considering strong imbalanced datasets: which is the most suitable metric to measure the performance of AI models? While the percentage of correctly classified instances is a standard metric for balanced datasets, there is no agreement over which metric or set of metrics is more suitable for multi-class imbalance classification problems. Therefore, a second objective of this research is to identify the most suitable metric for the evaluation of AI models in the classification of wind turbine failures. The most commonly used state-of-the-art values for the classification of imbalanced multiclass datasets are the Matthews Correlation Coefficient (MCC), the unweighted Macro average F (MacroF) and the geometric mean of the accuracy by class (G-mean) (Wang and Yao 2012; Wang et al. 2010; Joshi et al. 2010). There are also some studies which define certain extensions of the concept of Area Under the *Receiver Operating Characteristic Curve* (AUC) (Landgrebe and Duin 2008; Ferri et al. 2003), which is well-established for binary imbalance problems, but the formulation of this metric is not clear when it involves more than two dimensions. In our experiment, MacroF was chosen as the main metric and MCC was also used as complementary information. These and other popular metrics are detailed in Section "Metrics used for imbalanced classification", in which the reasoning is presented for considering MacroF as the most suitable approach to this problem. Finally, a similar problem arises when considering the classification techniques or AI models in the case of imbalanced datasets, especially as recent works have considered different approaches to imbalanced datasets, such as resampling or cost-sensitive learning.

The rest of this paper is organized as follows. The experimental procedure and the dataset used for the AI model training and validation are described in Section "Experimental procedure and data collection". Then, Sections "Metrics used for imbalanced classification" and "Multi-class imbalance classification" discuss the issues relating to data mining; the metrics for this kind of problem are detailed in Section "Metrics used for imbalanced classification" and the state-of-the-art classifiers are described in Section "Multi-class imbalance classification". Subsequently, the results of the AI modeling are discussed in Section "Results and discussion", including an in-depth analysis of the differences in AI model performance and the influence that the imbalance level of the dataset has on performance. Finally, in Section "Conclusions", the conclusions are presented along with future lines of research.

Experimental procedure and data collection

The test-bed set up was designed to simulate the behavior of wind turbines, while permitting testing under faulty operational conditions. It comprises an electrical drive (rather than a generator), a parallel gearbox and a planetary gearbox where the set of gears represents the configuration and the gear ratio of a commercial wind turbine of around 1:61, to simulate the powertrain of a real wind turbine, establishing the quick speed shaft of the test-bed. The test-bed is completed by a two-stage planetary gearbox and a brake that simulates the wind conditions with a random variation of rotating speed and load, establishing the slow speed shaft of the test-bed. The coupling between both parts of the test-bed is designed to allow controlled alignment and misalignment on both sides, thereby generating one of the two kinds of failures that will be tested: misalignment in the powertrain. Four ICP accelerometers, positioned in axial and radial directions from both gearboxes record vibrational signals. Both the current and the torque of the electrical drive of the test-bed were also recorded. Finally, the rotating speed was measured thanks to 24 holes performed in the blade to provide 24 keyphasors. In total, a 7-sensor system was built to monitor the test-bed performance. A complete list of the mechanical characteristics of the test-bed has been presented in a previous work (Villa et al. 2012). Figure 1 shows a scheme of the test-bed, with the quick speed shaft of the left and the slow one on the right, including the positions of all sensors.

The test-bed was programmed with random profiles of speed and load, in an attempt to simulate an extensive range of real working conditions. For this purpose, load and speed were changed, every 100 s, up to the next random condition. The selected speed range was between 1000 and 1800 rpm, and the load range was between 0 and 100 %, because that is the range at which the slow shaft of a wind turbine is able to generate energy. On average, each working condition was tested on the test-bed around 100 times during the measurement stage. The measurement time to capture the vibration

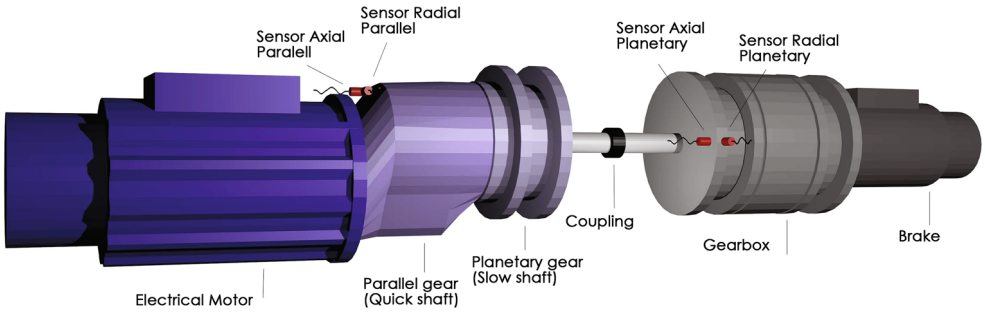


Fig. 1 Scheme of the test-bed

signal for each of the four accelerometers was fixed at 72 s with a sampling frequency (Δt) of 25,600 Hz.

After an in-depth analysis (Villa et al. 2012) of the precision and sampling frequency of the test-bed, similar to the ones that can be obtained in real wind mill generators, it was concluded that the vibrational signal will be distributed in a broad spectral range around any harmonic frequency, due to the change in rotating speed. Therefore, the application of a pre-processing stage of the acquired vibrational signals is compulsory, so that all of the information they collect may be extracted. The angular resampling methodology proposed by Fyfe and Munck (1997) was applied to these signals considering the specifications advanced by Villa et al. (2011). The angular resampling consists of: (1) an estimation of the resampled times $t * (k\Delta\theta)$, for which there is a constant angular increment; and (2) the fit of a spline interpolation of the original sampled vibration signal at the resampled times $t * (k\Delta\theta)$. The estimation of the resampled times $t * (k\Delta\theta)$ considers that the shaft turns with an constant angular acceleration, therefore a spline interpolation using the keyphasor angular positions $(\theta_1, \theta_2, \ldots \theta_k)$ and their arrival times $(t_1, t_2, \ldots t_k)$ is possible, as has previously been demonstrated in Villa et al. (2011). The angular increment is taken to have the original number of samples at the final angular position θ_k. Finally, a FFT of the resampled vibration signals was performed to translate these signals from the angular to the order domain.

Besides the misalignment of the powertrain, imbalance failures in the test-bed were also generated. Different levels of both kinds of failures were tested, to simulate progressive degradation of the wind turbine powertrain: two levels for misalignment and four levels for imbalance. Only in one case were both failures tested together, because this possibility is very uncommon in real wind farms. Altogether, seven failure cases and one non-failure case were tested. Table 1 shows the imbalance weight or the misalignment angle for each imbalance or misalignment failure case respectively. To simulate wind conditions, the test-bed ran under random profiles of speed and load covering an extensive range of real working conditions. Every 100 s, load and speed were changed to the next random condition. The programmed speed and load ranges were 1000–1800 rpm and 0–100 %, respectively. On average, each working condition was run on the test-bed 100 times. The accelerometer acquisition times were fixed at 72 s with a sampling frequency of 25,600 Hz. Rotation speed measurements simultaneously allow the use of the angular resampling technique on the vibration signals to extract useful information, regardless of the simulated variable wind condition (i.e., the rotation speed on the test-bed). This technique has been described in detail elsewhere (Villa et al. 2011). Each of these 72 s measurements generated an instance of the final dataset. In total, 6551 different working conditions were registered generating a dataset of 6551 instances. Table 1 also collects the percentage of measurements included in the final dataset for each failure type. We may conclude from Table 1 that the dataset was evenly balanced between the non-fault case and the seven fault cases; very clearly far removed from industrial reality.

Table 1 Distribution of measurements between the different failure instances that were tested

Type of fault	Number of instances
No misalignment–no imbalance	887 (13.54 %)
No misalignment–imbalance of 5.79 g	847 (12.93 %)
No misalignment–imbalance of 9.13 g	856 (13.07 %)
No misalignment–imbalance of 19.5 g	838 (12.79 %)
No misalignment–imbalance of 28.8 g	864 (13.19 %)
Misalignment of 0.78°–no imbalance	872 (13.31 %)
Misalignment of 1.53°–no imbalance	835 (12.75 %)
Misalignment of 1.53°–imbalance of 28.8 g	552 (8.43 %)

Each instance of the generated dataset is composed of many different variables. A first group of variables is associated with the operational state of the test-bed: torque, speed, electric input current and electric output current. A second group includes variables obtained from the analysis of the vibrational spectrum recorded by the accelerometers: some variables evaluate the energy distribution (average, root mean squared -RMS-, skewness, kurtosis and interquartile range), others take account of the energy allocated in standard fre-

Table 2 Variables included in the final dataset

Magnitude	Number of variables	Units	Range
Operation state			
Torque	1	% of maximum torque	1–100
Speed	1	rpm	1000–1800
Input current	1	A	1.63–2.85
Electrical current in the axis	4	mA	0.2–120
Vibration analysis			
Harmonics	272	$10^{-6} \times mm/s^2$	5.63–75
Bands	245	$10^{-6} \times mm/s^2$	16.5–52
Average	4	$10^{-3} \times mm/s^2$	−2 to 4
RMS	4	mm/s^2	0.016–0.12
Skewness	4	Dimensionless	1.40–4.78
Kurtosis	4	Dimensionless	2.19–66.1
Interquartile range	4	mm/s^2	0.021–0.17

quency bands or in the harmonics of the rotating speed. The total of 544 calculated variables are summarized in Table 2.

Metrics used for imbalanced classification

All the metrics considered in this research define an extension of measures which were previously formulated for binary imbalance classification problems. For that reason, the binary case is explained in this section before we present the generalization to multiple classes.

Binary problems

An imbalanced binary classification problem contains instances of two classes: one has the majority of instances, i.e., the *majority class* and the other has a few percentages of the instances, i.e., the *minority class*. So, the information that the classical metric provides to evaluate the performance of a classifier—the rate of correctly classified instances—is no longer reliable. For example, consider a dataset with 1000 instances of two classes: class A with 990 instances, and class B, with 10 instances. A dummy classifier that will always predict membership of class A for any instance, regardless of the class of that instance, would generate an expected accuracy of 99 %. However, this classifier would be unable to provide any useful information, as it would not have taken the *minority class* into account. In other words, class B might be undetectable with classifiers designed to solve general classification problems rather than a specific imbalance problem. In this context, the notation of the classes is changed to *positive class* (or *relevant information*) for the *minority class*, and *negative class* for the *majority class*, emphasizing the fact that an accurate estimation of the *positive class* is also needed, rather than an estimation of only the negative case.

Table 3 Confusion Matrix in binary problems, precision and recall

	Predicted value	
	Positive prediction	Negative prediction
Real value		
Positive instances	True positive (TP)	False negative (FN)
Negative instances	False positive (FP)	True negative (TN)

In our problem, the faults are *positive classes* (they occur with much less frequency than when the wind turbines are working properly).

Using the concepts of *positive* and *negative classes*, it is possible to distinguish between the type of mistakes that the forecast of a classifier could have, defining the so-called *confusion matrix* (Table 3), and, accordingly, the *precision (P)* and *recall (R)* values, as shown in Eqs. 1 and 2:

$$\begin{aligned} precision &= P = TP/(TP + FP) \\ &= TP/(positive\ prediction) \end{aligned} \tag{1}$$

$$\begin{aligned} recall &= R = TP/(TP + FN) \\ &= TP/(positive\ instances) \end{aligned} \tag{2}$$

Comparing the expressions that define both precision and recall, both may be seen to express a rate of true positives, but while precision is focused on the forecast values, recall is focused on the real values. In other words, with precision the target is expressed in terms of the reliability of the prediction (how many instances forecast as positives are actually positives?), while in the case of recall, the target is expressed in terms of sensitivity (how many positive instances can the classifier detect?). In both cases, the true positives are important, to arrive at accurate values of precision and recall.

However, good precision implies few false positives, while the number of false negatives should fall in the case of recall. When pondering on which is the most important, the question may be reformulated in terms of whether a false positive or a false negative is the worst mistake.

From the industrial point of view, the positive classes match faults in the machines and the negative class match proper operational conditions. So, if the situation which implies a higher cost is to stop the machine when a fault was not about to occur (i.e., in the case of a false positive), the classifier should target precision. On the other hand, if the worst scenario involves not stopping the machine when a fault is about to take place (i.e., in the case of a false negative), the aim should be to maximize recall.

In general, both precision and recall are important, so a set of metrics that combines both measures was defined. Two popular metrics for binary problems are explained: the F-measure and the MCC. Beside these measures, the AUC is also popular for binary problems, but its generalization to multi-class imbalance problems is not clear, for which reason it is not described in this work.

The F-measure is the harmonic mean of both precision and recall. Like the arithmetic mean, the harmonic mean provides us with the average of a set of numbers, although it uses inverse totals. This mean is defined as the inverse value of the mean of the inverses. In the case of the F-measure, the harmonic mean of precision and recall is given in Eq. 3. The main difference between the harmonic and the conventional mean is that low values strongly constrain the final average value in the first case. In fact, the F-measure would be zero when precision is one and when recall is zero, while in the case of having used an arithmetic mean it would be 0.5.

$$F = \frac{1}{\frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)} = \frac{2PR}{P+R} \tag{3}$$

Another way to estimate the performance of a classifier is by using the MCC, a metric based on statistical concepts. To define this measure, the data and the prediction of the model are seen as statistical variables, D and M, respectively. Then, the idea of correlation commonly used in statistics to define the rate of a relationship between two variables can be also used as a data-mining metric (Eqs. 4 and 5).

$$MCC = \frac{cov(D,M)}{\sqrt{cov(D,D)cov(M,M)}} \tag{4}$$

$$cov(X,Y) = mean[(X - mean(X))\,(Y - mean(Y))] \tag{5}$$

Once the two variables have been defined, D and M can be formulated into binary 0/1 vectors for the binary problems (the positive class being 1). In this particular case, the MCC can be more easily interpreted from the confusion matrix, the structure of which is illustrated in Table 4.

Table 4 Confusion Matrix in terms of D and M

	M	$\bar{M}$
D	TP	FN
$\bar{D}$	FP	TN

It is possible to calculate the covariance values in terms of the values of the confusion matrix (Baldi et al. 2000), as shown in Eqs. 6–8.

$$cov(D,M) = TP \cdot TN - FP \cdot FN \tag{6}$$

$$cov(D,D) = (TP + FN) \cdot (FP + TN) \tag{7}$$

$$cov(M,M) = (TP + FP) \cdot (FN + TN) \tag{8}$$

The final expression for the MCC in a binary problem is given in Eq. 9.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FN) \cdot (FP+TN) \cdot (TP+FP) \cdot (FN+TN)}} \tag{9}$$

Multi-class problems

Three metrics for multi-class problems are analysed in this work: the MacroF, the G-mean and the MCC. The metric taken as the main measure, the MacroF, generalizes the concept of the F-measure index to multiple dimensions, by averaging the F-measure values for each pair of classes, as shown in Eq. 10.

$$MacroF = \frac{\sum_{i=1}^{N} F - measure_i}{N} \tag{10}$$

The second multi-class metric analysed is the G-mean. This metric uses the geometric mean as an average of the accuracy obtained for each class, defined as the rate of correctly classified instances for this class (see Eq. 11). This concept is very similar to *recall* for the binary problem.

$$G - mean = \left[\prod_{i=1}^{N} \frac{Correct\ prediction\ i}{Total\ instances\ i}\right]^{1/N} \tag{11}$$

Finally, the MCC is detailed. The definition of this index for the multi-class problem is a current area of research (Jurman and Furlanello 2010). The formulation used in this work consisted of a weighted average of the MCC values obtained for each class (MCC_i). If there are m classes, N_i is the number of instances of the i-th class and N the number of total instances, then MCC is calculated as in Eq. 12, below. In this expression, if one class has almost every instance in the dataset (i.e., $N_i \rightarrow N$), then the MCC value tends to be equal to the value of the metric that was calculated for this majority

Table 5 Sensitivity to imbalance rate of the different approaches

	Classified as		
	OK	I	II
Case A (reference)[a]			
OK	995	5	0
I	5	45	0
II	0	0	10
Case B (asymmetric error in I)[b]			
OK	1000	0	0
I	10	40	0
II	0	0	10
Case C (bad performance in II)[c]			
OK	995	0	5
I	0	50	0
II	5	0	5

[a] MCC = 0.912, MacroF = 0.965
[b] MCC = 0.908 (⇓0.4 %), MacroF = 0.961 (⇓0.4 %)
[c] MCC = 0.912 (=), MacroF = 0.832 (⇓13.8 %)

class (i.e., $MCC \rightarrow MCC_i$). An example, in Table 5, was chosen to illustrate the differences between the MCC and the MacroF metrics.

$$MCC = \frac{\sum_{i=1}^{m} N_i \times MCC_i}{N} \tag{12}$$

In Table 5, an example test set of 3 classes was selected with 1000 instances of "OK" (the machine is working properly), 50 instances of failure type "I" and 10 instances of failure type "II". Three different confusion matrix cases (A, B and C) were proposed to compare the performance of the two metrics. Case A was taken as the reference, while B and C were used to see how the metrics were affected by changes in the error terms. The total of misclassified instances was the same in cases A and B (10 instances), but while in A the distribution of the error between classes "OK" and "I" was symmetric, in B one term of the confusion matrix has all the faults. Both the MCC and the MacroF values in confusion matrix B decreased by 0.4 % in comparison to confusion matrix A, so both metrics in the example have the same sensitivity to asymmetry in the error distribution. Case C was chosen to illustrate how the metrics were affected by poor performance in one specific minority class. While the MCC metric was not affected by changes in the confusion matrix, the variation in MacroF was considerable (13.8 %). It can be concluded that MacroF is the most suitable metric to evaluate the classification performance of a single class with a similar data distribution to our problem, because incorrect evaluation of one failure type in a wind turbine would never be acceptable from an industrial point of view. Rare failures in wind turbines can be much more critical than common failures and should be properly identified. In any case, the performance of the best classification models will be also evaluated with the MCC metric, which is a popular measure that facilitates comparisons between this research and the works of other authors. The information provided by MCC is also more convenient that the G-mean in our case, as optimization targets in terms of *precision* could be lost using the G-mean.

Multi-class imbalance classification

Multi-class imbalance classification has been applied to a variety of problems, such as bioinformatics (Tan et al. 2003), text classification (Rennie 2001), failure detection (Liao 2008) and image analysis (Montazer et al. 2012). However, no classification techniques are universally accepted as the most suitable for these kinds of problems. The methods presented in the bibliography may be categorized as follows:

1. Binarization ensemble (Bagheri et al. 2012; Hoens et al. 2012).
2. Cost-sensitive classifiers (Liu and Zhou 2006; Zhou and Liu 2010).
3. Resampling methods that balance the data (García and Herrera 2009; Estabrooks et al. 2004).
4. Classifiers for imbalanced binary problems (Lertampaiporn et al. 2013; Krawczyk and Schaefer 2013).

This categorization of the different techniques is useful to explain the general ideas that they share, although some further aspects should be added:

– Some works combine the techniques of different families. For example, it is possible to apply resampling first and then build a binarization ensemble.
– The fourth family belongs to the first one, but this division is more convenient for a clearer presentation, as this last type of classifier combines several approaches. They have gained popularity over recent years to solve imbalance classification problems, but only for binary cases and not for the kind of problem considered in this work, which belongs to binarization ensemble classification.

Binarization

This approach is probably the most commonly used to deal with this kind of classification, given that binary classification has been widely studied in the field of data mining. The initial problem of multiple classes is divided into a set of binary problems, with several ways of performing the decomposition. Then, an ensemble classifier is formed from the binary classifiers (i.e., base classifiers) (Hoens et al.

2012). These kinds of ensembles are designed to deal with multi-class problems *in general* and not specifically with *multi-class imbalance problems*. It is therefore more convenient to use them in combination with resampling techniques or with a base classifier capable of coping with imbalanced data. The most popular binarization techniques are 1-against-1, 1-against-All and Exhaustive Correction Code (ECOC) (Montazer et al. 2012).

In the 1-against-1 method (Hastie and Tibshirani 1998), a binary classifier is trained for each pair of classes, in contrast with the 1-against-All method (Anand et al. 1995), where the number of defined binary classifiers is only equal to the number of classes, taking the instances of one of the classes for each binary classifier and merging the instances from the rest of the classes into the other class. As a result, in a problem with k classes, $\binom{k}{2}$ problems are defined in 1-against-1, while only k are defined in 1 against All. In the present work, the sum of probability estimations for each class from the base classifiers is normalized to obtain the final probability estimations.

Unlike the two previous methods, in the case of ECOC (Dietterich and Bakiri 1995), the classes are not directly taken as the reference to define the binary problems, but they are previously codified in a binary code, taking one of the bits of this code for each class. Then, a binary problem is defined for each binary code where the two new classes are the instances from the original classes coded as 0 vs. the instances from the original classes coded as 1. Predictions from the base classifiers are concatenated to arrive at a binary code, finally assigning the class to the code with most similar bits to those that are generated. The idea is to generate a more robust system than other binarization techniques (Dietterich and Bakiri 1995), which can tolerate errors in the estimation of several bits without making a mistake in the final prediction. Specifically, if the minimum Hamming distance (number of different bits between codes) between two class codifications is d, the final ensemble is able to correct at least $(d-1)/2$ mistakes in individual bits.

Previous studies have proven the suitability of C4.4 and C4.5 decision trees as base classifiers for this type of ensemble, in the context of a general multi-class problem (Fürnkranz 2002). However, the problem under study in this work not only involves multi-class but also imbalanced databases. For that reason, besides separately considering these classifiers, they are combined with a resampling technique and are also used as base classifiers within specific binary-imbalanced ensembles (SMOTEBagging, SMOTEBoosting and RUSBoost). In the experiments, besides decision trees, Linear SVM was also included as a possible base classifier for binarization ensembles.[1]

[1] SVM is in fact a binary classifier, but it is commonly used as multi-class classifier using 1-against-1.

Cost-sensitive classifiers

In classification problems with imbalance, the most common way to evaluate the performance of a classifier, the rate of correctly classified instances, is not suitable (Pazzani et al. 1994). Other metrics are instead defined, as explained in Section "Metrics used for imbalanced classification". Here, we encounter the concepts of *misclassification cost* between two classes and *cost matrix* for each pair of classes. The misclasification cost of classifying an instance as the i^{th} class, which is in reality of the $j-th$ class, is referred to as $cost(i,j)$. It is important to emphasize that it is not a symmetrical operator; $cost(i,j) \neq cost(j,i)$. Medical diagnosis may be taken as an example to explain this concept. It is not acceptable to compare, in the same terms, a notification that informs a person with a disease that the person is healthy and a communication that mistakenly informs a healthy person of an illness.

Breiman et al. (1984) introduced a method to train the classifier previously modifying the dataset to take into account the misclassification costs. Specifically, the *a priori* probabilities of each class are reassigned using the expression of Eq. 13, where $\Pi(k)$ is the *a priori* probability of the class k-th and $C(k)$ its total cost.

$$\Pi(j) = \frac{C(j)\Pi(j)}{\sum_i C(i)\Pi(i)}\ ,\ where\ C(j) = \sum_i cost(j,i) \quad (13)$$

A limitation of this procedure is that it demands a priori knowledge of the misclassification cost between each pair of classes. In this work, the choice of this cost has been systematized, taking the inverse of the class frequencies as a reference, in order to penalize forecasting the majority class. $cost_{ij}$ is zero for the terms of the main diagonal of the cost matrix (when $i = j$), and in the other cases Eq. 14 should be satisfied.

$$cost_{ij} = \frac{C_i}{C_j} = \frac{frequency_j}{frequency_i} \quad (14)$$

To complete Eq. 14, the sum of all the coefficients in the cost matrix should be equal to the number of classes.

Resampling-based classifiers

Resampling methods modify the dataset by obtaining a balanced problem from the original imbalanced situation (He and Garcia 2009). Given a dataset in which one set of classes consists of majority classes (one in the studied problem) and the other consists of minority classes (seven in the case of study), there are two ways of getting the same frequency for all classes:

- UnderSampling: taking a reduced set of the instances of the majority classes to generate a similar number of instances to the minority.
- OverSampling: artificially generating more instances of the minority classes to obtain a similar number to the majority classes.

In this work, a set of instances of the majority class was randomly eliminated for the case of UnderSampling, while in the case of OverSampling the Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al. 2011) was used. In this method, a set of instances is taken from each of the minority classes. We call each these instances $x_{initial}$. We also consider a further instance of the same class for each $x_{initial}$, with values of greater similarity to the attributes (the nearest neighbours in the attributes space), that we call $x_{neighbour}$. A new instance situated between both points is computed for each pair $(x_{initial}, x_{neighbour})$, which is calculated by using a random number between 0 and 1, according to the formula $x_{new} = x_{initial} + (x_{neighbour} - x_{initial}) \times random$.

Imbalanced binary classification

Unlike in the imbalanced multi-class classification, in the imbalanced binary classification, there is a greater consensus over which approaches are the most acceptable to deal with the imbalance (Galar et al. 2012). Four popular classifiers from among these techniques were used as base classifiers of a binarization ensemble (4.1) that was tested in this work:

- C4.4 (Provost and Domingos 2003). This technique is a modification of C4.5 decision trees (Quinlan 1993), in which pruning and collapsing are eliminated and the class probabilities are calculated with the Laplace correction. It is one of various methods that *smooth probability estimates from small samples* (Simonoff 1995), thereby improving the estimation of probabilities given by decision trees, as these kinds of classifiers are not considered good probability estimators (Bradley 1997).
- SMOTEBagging (Wang and Yao 2009): Bagging ensemble (Breiman 1996), having previously balanced the dataset with SMOTE. Not all the initial instances of the majority classes are taken, but they are resampled.
- SMOTEBoost (Chawla et al. 2003): Adaboost ensemble (Freund et al. 1996), having previously balanced the dataset with SMOTE. As in the case of SMOTEBagging, a resampling of the instances of the majority classes is performed.
- RUSBoost (Seiffert et al. 2010): Adaboost ensemble, having previously balanced the data with Random Under Sampling.

Results and discussion

This section has been split into four subsections for the sake of clarity. In the first, the experimentation procedure is presented; in the second, the values of the metric MacroF for the different groups of classifiers; in the third, the focus is on the sensitivity of each group of classifiers to the degree of dataset imbalance and in the last section, we identify the best AI model for this industrial task and extract some overall conclusions.

AI experimental procedure

A 5×2 cross validation (Kohavi 1995) was performed using WEKA (Witten and Frank 2005). As explained in Section "Metrics used for imbalanced classification", the MacroF metric was taken as the reference, also indicating some information on the MCC. Besides the four families of techniques explained in Section "Multi-class imbalance classification", two further cases were considered: the individually tested classifiers and a combination of families 4.1 and 4.3 (binarization ensemble of binary classifiers and resampling). However, the experimental procedure was divided into 8 parts in total rather than into 6. Due to the large number of resampling cases, it was considered more convenient to distinguish between *oversampling* and *undersampling*, not only for this approach, but also in the case where it is combined with binarization ensembles of binary classifiers. The 8 cases are as follows:

- Undersampling (Under)
- Oversampling (Over)
- Binarization ensemble of binary not imbalance-oriented classifiers (Bin NOT Imb)
- Binarization ensemble of binary-imbalance classifiers (Bin Imb)
- Binarization ensemble + Undersampling (Bin + Under)
- Binarization ensemble + Oversampling (Bin + Over)
- Cost-sensitive classifiers (Cost)
- Classifiers without a specific multi-class imbalance technique (Raw)

Table 6 summarizes the configuration used for each experimentation. The set of base classifiers considered in the groups indicated in the second row of Table 6 is as follows:

- The *binary* group of classifiers consisted of C4.5, C4.4 and Linear SVM. While SVM is a binary classifier, the performance of decision trees can be improved with an ensemble of classifiers trained in binary datasets formed from the original ("binarization ensemble"), as explained in section "Binarization".

Table 6 Configuration of each experiment

Under	Over	Bin + NOT Imb	Bin + Imb	Bin + Over	Bin + Under	Cost	Sing
General	General	Binary	Binary−Imbalance	Binary	Binary	General	General

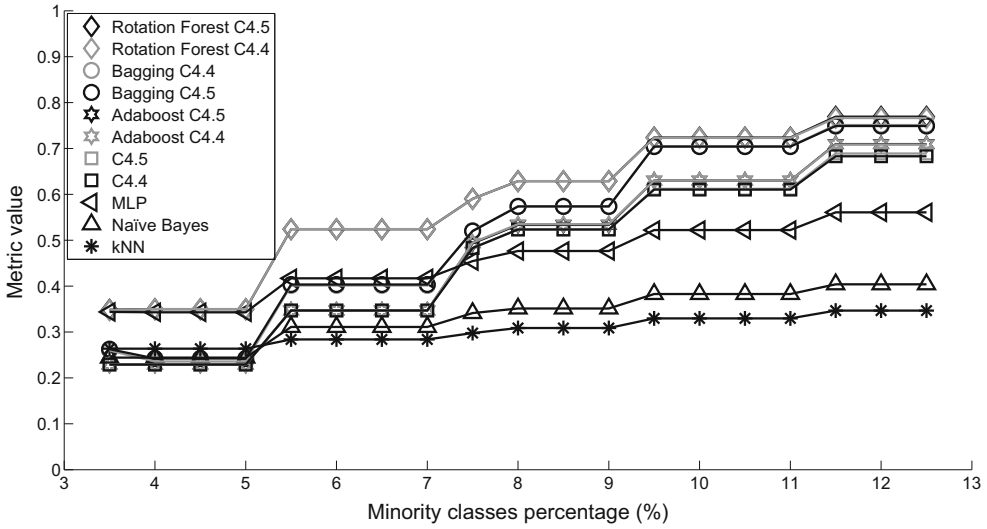


Fig. 2 MacroF − undersampling

– In the *general* group of classifiers, a variety of the most popular data-mining techniques from various approaches were used. These classifiers were: Neural Network (Multi-Layer Perceptron, MLP) (Hornik et al. 1989), Naïve Bayes (NB) (John and Langley 1995), *k*-Nearest Neighbour Classifier (kNN) (Vijayakumar and Schaal 2006), Decision Tree (C4.5) (Quinlan 1993), and three Decision Tree ensembles: Bagging (Breiman 1996), Adaboost (Freund et al. 1996) and Rotation Forest (Rodríguez et al. 2006). In an ensemble classifier the predicted class is obtained from the combination of several classifiers, called *base classifiers* , using a voting system. In the three ensemble classifiers considered in this study each base classifier uses all the attributes to obtain its individual prediction. Other possibilities with base classifiers specialised in a set of the attributes were not taken into account (Modi et al. 2011). The most widely referenced procedure to obtain the weights for this sort of ensemble is based on minimizing an error function, but the traditional error function for classification is optimal in the sense of accuracy and not in the sense of an specific multi-class imbalanced metric.
– The *binary-imbalance* group of classifiers consisted of SMOTEBoost, RusBoost and SMOTEBagging, as explained in Section 4.4. These were not combined with the resampling techniques, because they automatically perform resampling.

Several datasets were formed from the original data with different degrees of minority class imbalance. The number of instances that belong to any minority classes (those that represent faults of the wind turbine) varied in these data sets from 3 to 12.5 % of the total number of instances. This range of imbalance seeks to simulate the real conditions of data sets from the wind-turbine industry. Indeed, the lower limit of 3 % was fixed, because some undersampling techniques will not work under this limit (due to an insufficient number of instances of some classes).

Analysis of MacroF by groups of classifiers

Figures 2, 3, 4, 5, 6, 7, 8 and 9 show the evolution of the MacroF obtained by the classifiers as the percentage of the minority classes increases in the eight experiments described above. The same scale was used to allow a direct graphical comparison of the MacroF level reached by each method. For all the ensembles in the figures, the methods based on the same ensemble technique but with different kinds of trees as their base classifier are shown with the same label: black for the C4.4 tree and grey for the C4.5 tree. In the legends of the figures, the methods are listed by their MacroF value in descending order, at the higher minority classes rate (12.5 %).

Regarding the best and worst methods of each family, the following facts are shown in Figs. 2, 3, 4, 5, 6, 7, 8 and 9:

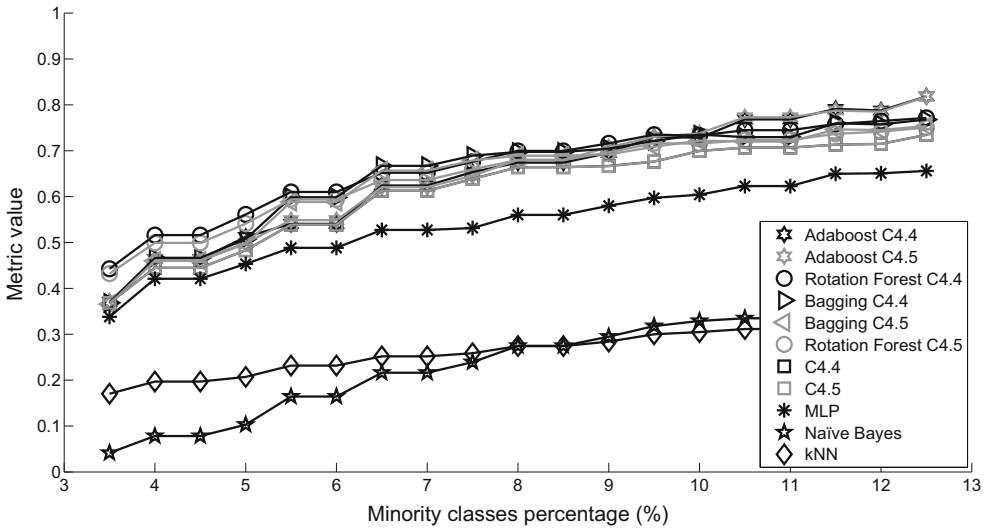


Fig. 3 MacroF − oversampling

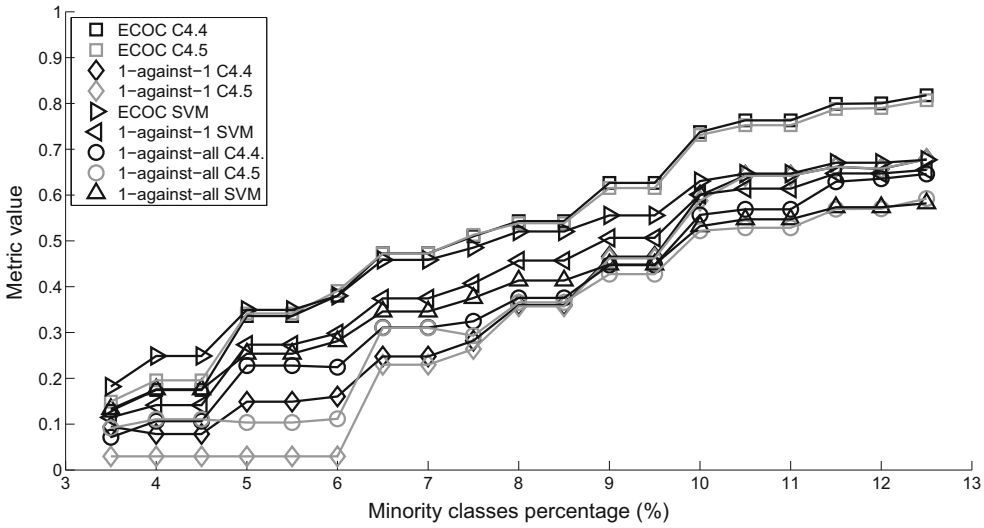


Fig. 4 MacroF − binarization ensemble + binary NOT imbalance-oriented

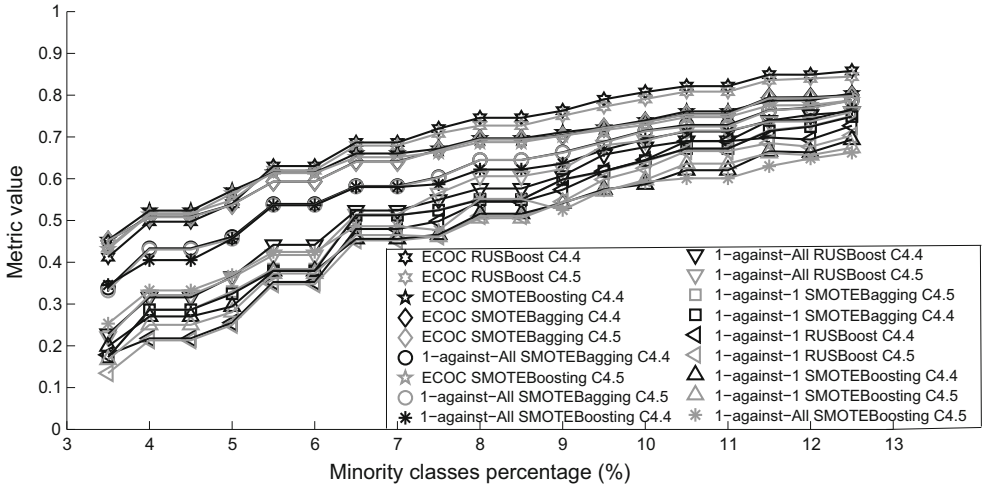


Fig. 5 MacroF − binarization ensemble + binary imbalance classifiers

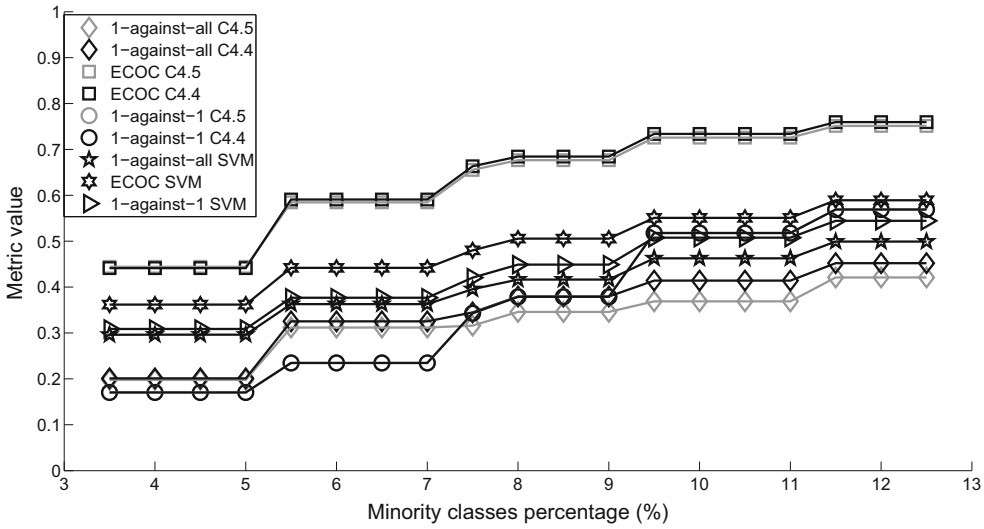


Fig. 6 MacroF − binarization ensemble + undersampling

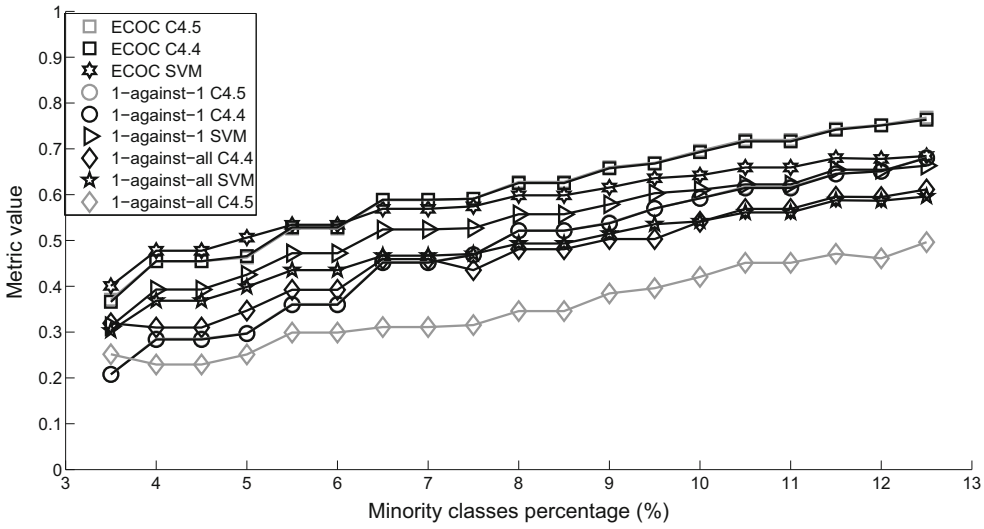


Fig. 7 MacroF − binarization ensemble + oversampling

– UnderSampling (Fig. 2)
  – The best cases were Rotation Forest for both C4.4 and C4.5 at any rate of imbalance. MLP and Bagging (with both C4.4 and C4.5) also performed well, but only for some specific imbalance rates. With a minority classes rate lower than 5.5 %, MLP fell within the group of top-ranked MacroF metrics, while the Bagging classifier could be included among the best methods, when the minority classes rate was higher than 9.5 %.
  – The worst algorithms were Naïve Bayes and kNN. They clearly showed the worst performance of all of the classifiers when the minority classes rate was higher than 5 %.
– OverSampling (Fig. 3)
  – At a low degree of imbalance, the best cases were Adaboost (with both C4.4 and C4.5). However, the best method was Rotation Forest (with both C4.4 and C4.5) at a high degree of imbalance.
  – kNN and Naïve Bayes clearly had the worst performance of all of the classifiers.
– Binarization ensemble + Binary NOT imbalance-oriented (Fig. 4)

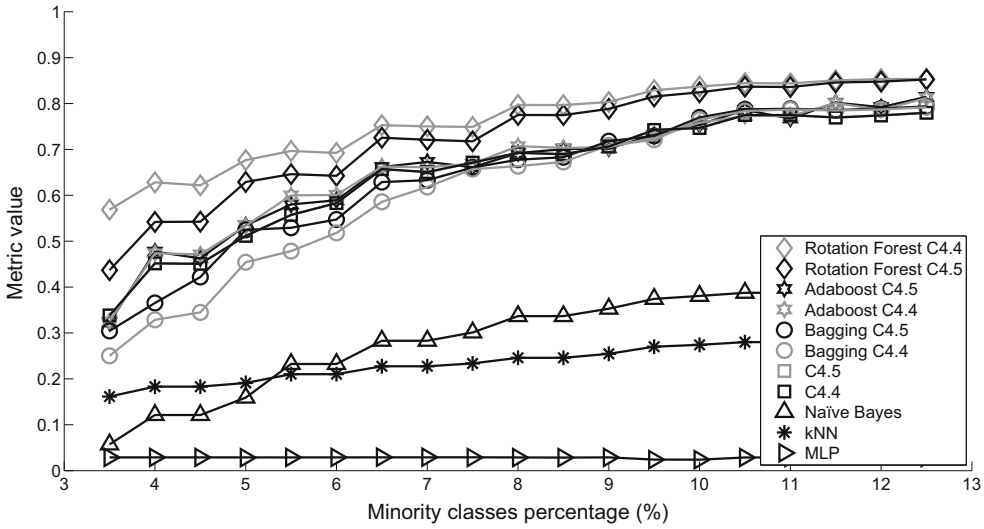


Fig. 8 MacroF – cost-sensitive

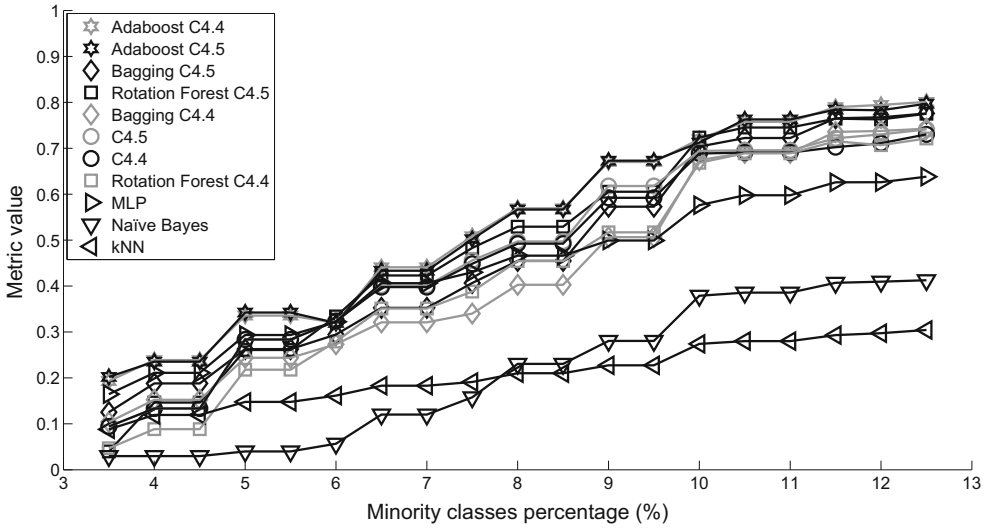


Fig. 9 MacroF – raw classifiers

- The best cases were ECOC with both C4.4 and C4.5 at any imbalance rate and ECOC SVM with a minority classes rate of below 5 %.
- The worst algorithm was 1-against-1 C4.5 with a minority classes rate of below 6.5 %.

– Binarization ensemble + Binary-imbalance (Fig. 5)

- The best methods were ECOC RUSBoost with both C4.5 and C4.5 with a minority classes rate of over 6 %. One group of classifiers had a similar performance with lower rate (ECOC SMOTEBoosting C4.4 had a higher MacroF in this case).
- No single group of classifiers clearly had a lower performance than the others.

– Binarization ensemble + UnderSampling (Fig. 6)

- The best methods were ECOC with both C4.4 and C4.5.
- 1-against-1 and 1-against-all ensembles of trees (C4.4 and C4.5) were the worst classifiers. They clearly showed a worst performance than the rest of the classifiers, when the minority classes rate was lower than 9.5 %.

– Binarization ensemble + OverSampling (Fig. 7)

- The best cases were ECOC with both C4.5 and C4.5 at any imbalance rate and ECOC SVM with a minority classes rate lower than 8 %.

  - The worst algorithm was 1-against-All C4.5 at any imbalance rate except 3.5, where 1-against-1 C4.5 was worst.

- Cost-sensitive (Fig. 8)
  - The best methods were Rotation Forest with C4.4 and C4.5 as base classifiers. Using C4.4 instead of C4.5 provided a clear advantage with a high imbalance (minority classes rate lower than 6 %).
  - The worst classifiers were Naïve Bayes, kNN and finally MLP, which was clearly the weakest of the three.
- Raw classifiers (Fig. 9)
  - The best methods were Adaboost with both C4.4 and C4.5.
  - The worst classifiers were kNN and Naïve Bayes. With a minority classes rate lower than 5 %, Rotation Forest (with both C4.4 and C4.5) also fell within the group of low-ranked methods.

Analysis of sensitivity-to-imbalance level

Table 7 shows the extent to which the MacroF value reached by the best classifiers was influenced by the imbalance rate, through the calculation of the number of classifiers that reached a MacroF value of 0.5 at different levels. This analysis shows that a cost-sensitive classifier of the Rotation Forest ensemble with C4.4 as a base classifier was the most robust method regarding the imbalance level, as its MacroF was never lower than 0.5. The other five classifiers that were stabler than the rest are as follows:

- Binarization ECOC ensemble of RUSBoost with C4.4 base classifiers.
- Binarization ECOC ensemble of SMOTEBoosting with C4.4 base classifiers.
- Rotation Forest ensemble of C4.4 base classifiers combined with Oversampling.
- Adaboost ensemble of C4.4 base classifiers with combined with Oversampling.
- Binarization ECOC ensemble of Linear SVM combined with Oversampling.

Table 7 Sensitivity to imbalance rate of the different approaches

Minority classes rate	Number of methods achieving $MacroF \geq 0.5$
<4 %	Only 1
4–4.5 %	6
>5.5 %	All (14)

Identification of the best classifier and general conclusions

Finally, after having performed the analysis for the eight experiments, two methods for each case were selected for an in-depth exploration of the differences between the approaches: the classifier which maximizes the MacroF at both the higher and the lower rate of the minority classes (when a classifier is the winner in both situations, only one is chosen for this family). The results of the selection are described below:

- Between undersampling classifiers: Rotation Forest of C4.5.
- Between oversampling classifiers: Rotation Forest of C4.4 and Adaboost of C4.4.
- Between binarization ensembles of binary not imbalanced-oriented classifiers: ECOC with C4.4 and ECOC with Linear SVM
- Between binarization ensembles of binary imbalance-oriented classifiers: ECOC with C4.4 SMOTEBoosting and ECOC with C4.4 RUSBoost.
- Between binarization ensembles with undersampling: ECOC with C4.4 and ECOC with C4.5.
- Between binarization ensembles with oversampling: ECOC with C4.5 and ECOC with Linear SVM
- Between cost-sensitive classifiers: Rotation Forest of C4.4.
- Between raw classifiers: Adaboost of C4.4 and Adaboost of C4.5.

In Fig. 10, the MacroF values of the best methods are compared. As this figure shows, a cost-sensitive classifier with a Rotation Forest ensemble of C4.4 was the best classifier found in the experiments in terms of MacroF. It obtained the best metric value in all the percentages of the minority classes that were tested.

Finally, in Table 8 the numeric values of MacroF for the best methods are presented, sorted by decreasing average value and with the highest values highlighted in bold. An asterisk indicates the statistically significative cases that were worse than the best classifier: the Cost-Sensitive classifier with a Rotation Forest ensemble of C4.4, using a resampled t-test (Nadeau and Bengio 2003) with a 95 % confidence level.

The following conclusions represent the general results for the top-ranked methods in terms of MacroF:

- Cost-sensitive classification (Fig. 8) and binarization ensembles of binary-imbalance classifiers (Fig. 5) were better approaches to these multi-class imbalance problems than strategies that used resampling outside the classifier or classifiers without a specific multi-class technique.

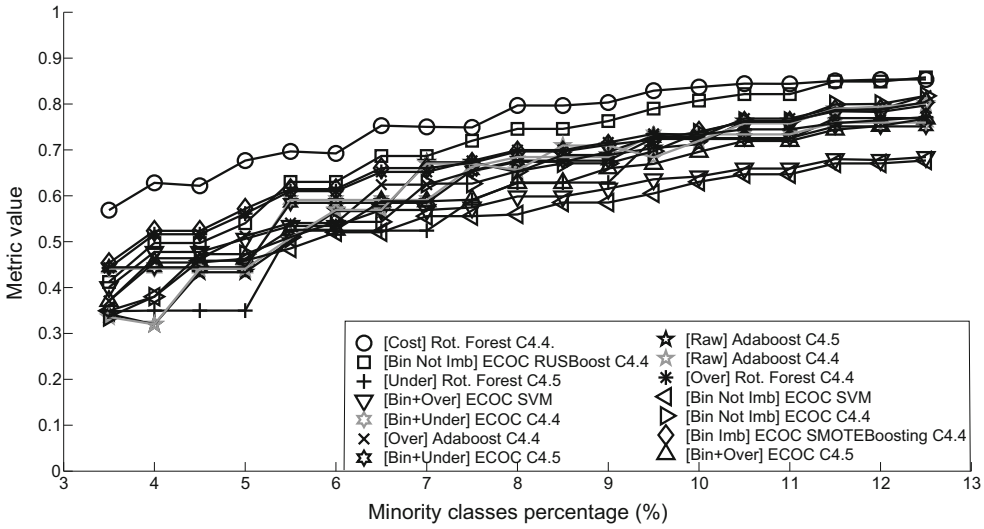


Fig. 10 MacroF – Best methods

Table 8 MacroF of the best methods varying imbalance level from 3.5 to 12.5 %

Method/Imb.	3.5	4.5	5.5	6.5	7.5	8.5	9.5	10.5	11.5	12.5
(Cost) R. Forest C4.4.	**0.57**	**0.62**	**0.70**	**0.75**	**0.75**	**0.80**	**0.83**	**0.84**	**0.85**	0.85
(Bin Imb) EC. RUSBo. C4.4	0.41	0.50	0.63	0.69	0.72	0.75	0.79	0.82	**0.85**	**0.86**
(Bin Imb) EC. SMOTEBo. C4.4	0.45	0.52	0.61*	0.66*	0.67	0.70*	0.72*	0.76*	0.79*	0.80*
(Over) R. Forest C4.4	0.44*	0.52	0.61	0.65*	0.68	0.70*	0.74*	0.74*	0.76*	0.77*
(Over) Adaboost C4.4	0.37*	0.46*	0.54	0.62*	0.65*	0.67	0.73	0.77	0.79	0.82
(Bin + Under) ECOC C4.4	0.44	0.44*	0.59	0.59	0.66	0.68*	0.73	0.73*	0.76*	0.76*
(Bin + Under) ECOC C4.5	0.44	0.44	0.58	0.58*	0.66	0.68*	0.73*	0.73*	0.75*	0.75*
(Raw) Adaboost C4.4	0.34*	0.44*	0.51*	0.57*	0.67	0.71	0.68	0.76	0.79	0.80
(Raw) Adaboost C4.5	0.34*	0.43*	0.50*	0.57*	0.67	0.69*	0.70	0.76	0.78	0.80
(Bin Not Imb) ECOC SVM	0.34*	0.43*	0.50*	0.57*	0.67	0.69*	0.70	0.76	0.78	0.80
(Bin + Over) ECOC C4.5	0.37*	0.45*	0.53	0.59*	0.59*	0.63*	0.67*	0.72*	0.74*	0.77
(Under) R. Forest C4.5	0.35*	0.35*	0.52*	0.52*	0.59	0.63*	0.72*	0.72*	0.77	0.77
(Bin + Over) ECOC SVM	0.40*	0.48*	0.53*	0.57*	0.57*	0.60*	0.64*	0.66*	0.68*	0.68*
(Bin Not Imb) ECOC C4.4	0.24*	0.32*	0.34*	0.40*	0.51*	0.58*	0.62*	0.69*	0.74*	0.74*

* Statistically significative worst cases

- Within the top-ranked methods, C4.4 was a better base classifier than C4.5.
- Within the binarization approaches (Figs. 4, 5, 6, 7), a combintion of the binary classifier via ECOC outperformed 1-against-1 and 1-against-All.
- In two cases, the values of the metric MacroF were very poor. This could have been caused by the well-know problem of overfitting. (Quinlan 1993). These situations are 1-against-1 C4.5 in the binarization ensembles of the binary not imbalance-oriented classifiers (Fig. 4) (i.e., 0.030 in the minority classes range of 3.5–6 %) and MLP inside a cost-sensitive classifier (about 0.028) (Fig. 8).
- There are specific differences in Figs. 2 and 6, which match families with undersampling, because it is possible to see *steps* in the figures. These approaches show less sensitivity to the level of imbalance, because the training sets tended to be more similar, as they contained a lower number of training instances.
- When the minority classes rate was higher than 10 %, the rising curve became smoother (i.e., the performance of the methods was stabler). This means that if the data set includes a high number of instances describing failures—a difficult industrial task—the differences in performance between the classifiers will be less significant.
- When the level of imbalance was very high, the value of the MacroF metric was poor. Taking into account that 0.5 is the standard value for a random prediction, it was possible to set a minimum reference value of 0.75 to analyze an acceptable level of imbalance for the classifiers. The

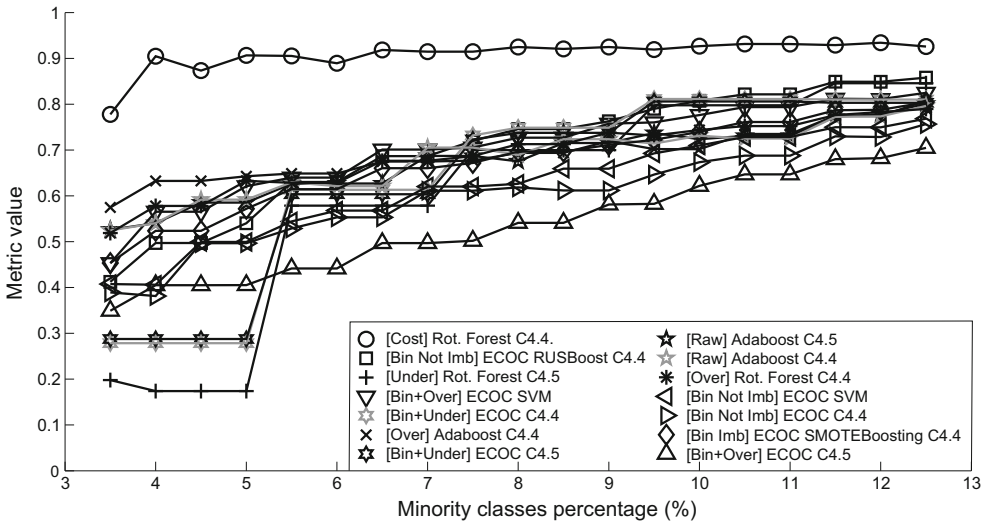


Fig. 11 MCC – Best methods

best method (Cost-sensitive classifier of Rotation Forest with C4.4 decision tree as the base classifier) could reach this reference when the minority classes rate was 6.5 %, while the second method (Binarization ECOC ensemble of Rusboost with a C4.4 base classifier) needed a rate of 8 %. For the rest of the classifiers, no one classifier reachesed the reference until the minority classes rate was 10.5 %.

– The MacroF metric for the two best methods took a value of around 0.85 at the end of the curves. This can be considered a good performance, as the problem that involved 8 classes was still very imbalanced with this minority classes rate.

The information on the MacroF metric is complemented with the MCC metric in Fig. 11. We may also conclude from this measure that the two best methods were a cost-sensitive classifier of the Rotation Forest ensemble with C4.4 as a base classifier and a binarization ensemble of RusBoost C.4.4 base classifiers, combined with the ECOC technique. The first classifier obtained the highest MCC for all percentages of the minority classes except the last one (12.5 %).

Conclusions

An extensive dataset generated in real tests on a test-bed has been described in this paper, in an attempt to answer questions of industrial interest for the implementation of intelligent system diagnostics in wind turbines: firstly, which is the best metric to measure the performance of AI models working with imbalanced datasets and, secondly, which is the best AI model to classify machine states in a wind generator, depending on the level of imbalance. The experimental tests included information from accelerometers, electrical conversion, torque and rotation speed. An angular resampling technique was applied to the accelerometers signals, to analyze the effect of variable wind speed and load. The final data set included 6551 instances classified in 8 cases: 7 failure cases and 1 non-failure cases. Each instance was composed of 544 inputs that included global variables associated with the operational state of the test-bed and an extensive set of variables obtained from the analysis of the vibrational spectrum of the accelerometers. The dataset was evenly balanced between the non-fault case and the seven fault cases, a situation clearly far removed from industrial reality. Some datasets are formed from the original, varying the total rate of the instances of the minority classes, to imitate a situation comparable to the real operation conditions of wind turbines. The type of data-mining problem under analysis was therefore a multi-class imbalance classification.

Several metrics for multi-class imbalance classification were studied to answer the first research question, to conclude that MacroF is the most suitable metric to analyze the performance of the classifiers from an industrial point of view. As the MCC metric is very commonly used, its calculation might be of interest to facilitate comparison between different research works. The experimental test shows that MacroF rather than MCC can better illustrate the loss of confidence in the results of the classification models, when the level of imbalance in the data set is increased.

In answer to the second question, several families of classifiers of the state of the art for multi-class imbalance classification were compared. The results led us to conclude that the best approaches were cost-sensitive classifiers and binarization ensembles of binary classifiers specialized in the binary imbalanced problem. Specifically, a cost-sensitive classifier of a Rotation Forest ensemble with C4.4 as base

classifier was the best method found in the experimentation. High values (0.75) of the MacroF metric were obtained with this classifier, when the minority classes rate fell within the 6.5–13 % interval, while the majority of the classifiers were unable to achieve good performance until the minority classes rate was 10.5 %. The Cost-Sensitive Rotation Forest ensemble of C4.4 was clearly the best prediction model below an imbalance level of 6.5 % (interval 3.5–6.5 %), although its performance under industrial conditions would require further testing, because although MCC remained higher than 0.85, the MacroF metric was in the range of 0.58–0.75.

Future work will focus on the application of this methodology to other types of wind-turbine failures, so that the same software diagnostics system may be tested under a wider range of operational conditions. Another new line of research could be the programming of ensembles specifically designed to solve multi-class imbalanced problems. A new search procedure to calculate the weights of each base classifier has to be defined, in order to obtain these new ensembles, in such a way that the MacroF is maximized.

Acknowledgments This research project has received funding from the Spanish government through Projects CENIT-2008-1028, TIN2011-24046 and IPT-2011-1265-020000 of the Ministerio de Economía y Competitividad [Ministry of Economy and Competitiveness]. Special thanks to Roberto Arnanz, Dr. Luisa F. Villa and Dr. Aníbal Reñones of the CARTIF FOUNDATION for providing the original dataset and for performing all the experimental tests and to Dr. Juan J. Rodríguez from the University of Burgos for his kind-spirited and useful advice.

References

Anand, R., Mehrotra, K., Mohan, C. K., & Ranka, S. (1995). Efficient classification for multiclass problems using modular neural networks. *Neural Networks, IEEE Transactions on*, *6*(1), 117–124.

Bagheri, M. A., Montazer, G. A., & Escalera, S., (2012). Error correcting output codes for multiclass classification: application to two image vision problems. In *2012 16th CSI international symposium on artificial intelligence and signal processing (AISP)* (pp. 508–513). IEEE.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics*, *16*(5), 412–424.

Barszcz, T., & Randall, R. B. (2009). Application of spectral kurtosis for detection of a tooth crack in the planetary gear of a wind turbine. *Mechanical Systems and Signal Processing*, *23*(4), 1352–1365. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0888327008002239

Bartelmus, W., & Zimroz, R. (2009). Vibration condition monitoring of planetary gearbox under varying external load. *Mechanical Systems and Signal Processing*, *23*(1), 246–257, special Issue: Non-linear Structural Dynamics. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0888327008000824

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. In *Wadsworth International Group*.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, *24*(6), 2350–2383.

Bustillo, A., & Correa, M. (2012). Using artificial intelligence to predict surface roughness in deep drilling of steel components. *Journal of Intelligent Manufacturing*, *23*(5), 1893–1902.

Cao, Y. H., Cao, Y., Wu, G. Q., Li, Q. M., & Shi, Y. J. (2014). The analysis of monitoring system of wind turbine. *Applied Mechanics and Materials*, *487*, 595–600.

Caselitz, P., Giebhardt, J., & Mevenkamp, M. (1994). On-line fault detection and prediction in wind energy converters. In *Proceedings of the EWEC* (Vol. 94, pp. 623–627).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2011). Smote: Synthetic minority over-sampling technique. arXiv preprint arXiv:1106.1813.

Chawla, N. V., Lazarevic, A., Hall, L. O., & Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In N. Lavrač, D. Gamberger, L. Todorovski, & H. Blockeel (Eds.), *Knowledge discovery in databases: PKDD 2003* (pp 107–119). Springer.

Chen, J., & Hao, G. (2012). Research on the fault diagnosis of wind turbine gearbox based on bayesian networks. *Practical Applications of Intelligent Systems*, *124*, 217–223.

Davies, A. (1998). *Handbook of condition monitoring: Techniques and methodology*. Chapman & Hall. [Online]. Available http://books.google.es/books?id=j2mN2aIs2YIC

Dietterich, T. G., & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. arXiv:cs/9501101.

Essawy, M. (1998). Fault diagnosis of helicopter gearboxes using neuro-fuzzy techniques. In *52nd meeting of the MFPT society*, pp. 293–302.

Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, *20*(1), 18–36.

Ferri, C., Hernández-Orallo, J., & Salido, M. A. (2003). Volume under the roc surface for multi-class problems. In *Machine learning: ECML 2003* (pp. 108–120). Springer.

Filev, D. & Yager, R. R. (1994). Learning owa operator weights from data. In *Fuzzy systems, 1994. IEEE World congress on computational intelligence. Proceedings of the third IEEE conference on*, (pp. 468–473). IEEE.

Filev, D., & Yager, R. R. (1998). On the issue of obtaining owa operator weights. *Fuzzy sets and systems*, *94*(2), 157–169.

Freund, Y., & Schapire, R. E. et al. (1996). Experiments with a new boosting algorithm. In *ICML* (Vol. 96, pp. 148–156).

Fürnkranz, J. (2002). Round robin classification. *The Journal of Machine Learning Research*, *2*, 721–747.

Fyfe, K., & Munck, E. (1997). Analysis of computed order tracking. *Mechanical Systems and Signal Processing*, *11*(2), 187–205.

Gajate, A., Haber, R., del Toro, R., Vega, P., & Bustillo, A. (2012). Tool wear monitoring using neuro-fuzzy techniques: A comparative study in a turning process. *Journal of Intelligent Manufacturing*, *23*(3), 869–882.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, *42*(4), 463–484.

García, S., & Herrera, F. (2009). Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary Computation*, *17*(3), 275–306.

Garg, A., & Tai, K. (2014). An ensemble approach of machine learning in evaluation of mechanical property of the rapid prototyping fabricated prototype. In *Applied Mechanics and Materials* (Vol. 575, pp. 493–496). Trans Tech Publ.

Hameed, Z., Hong, Y., Cho, Y., Ahn, S., & Song, C. (2009). Condition monitoring and fault detection of wind turbines and related algorithms: A review. *Renewable and Sustainable energy reviews*, *13*(1), 1–39.

Harris, T. (1993). A kohonen som based, machine health monitoring system which enables diagnosis of faults not seen in the training set. In *Neural networks, 1993. IJCNN'93-Nagoya. Proceedings of 1993 international joint conference on*, (Vol. 1, pp. 947–950) IEEE.

Hastie, T., & Tibshirani, R. (1998). Classification by pairwise coupling. *The Annals of Statistics*, *26*(2), 451–471.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, *21*(9), 1263–1284.

Hoens, T.R., Qian, Q., Chawla, N. V., & Zhou, Z.-H. (2012). Building decision trees for the multi-class imbalance problem. In P.-N. Tan, S. Chawla, C. K. Ho, & J. Bailey (Eds.), *Advances in knowledge discovery and data mining* (pp. 122–134). Springer.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366.

Jeffries, W., Chambers, J., & Infield, D. (1998). Experience with bicoherence of electrical power for condition monitoring of wind turbine blades. In *IEE proceedings—vision, image and signal processing* (Vol. 145, no. 3, pp. 141–148). IET.

John, G. H., & Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the eleventh conference on uncertainty in artificial intelligence* (pp. 338–345). Morgan Kaufmann Publishers Inc.

Joselin Herbert, G., Iniyan, S., Sreevalsan, E., & Rajapandian, S. (2007). A review of wind energy technologies. *Renewable and Sustainable Energy Reviews*, *11*(6), 1117–1145.

Joshi, A. J., Chandran, S., Jayaraman, V. K., & Kulkarni, B. D. (2010). Hybrid support vector machine for imbalanced data in multiclass arrhythmia classification. *International Journal of Functional Informatics and Personalised Medicine*, *3*(1), 29–47.

Jurman, G., & Furlanello, C. (2010). A unifying view for performance measures in multi-class prediction. arXiv preprint arXiv:1008.2908.

Kohavi, R., et al. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *IJCAI*, *14*(2), 1137–1145.

Krawczyk, B., & Schaefer, G. (2013). An improved ensemble approach for imbalanced classification problems. In *2013 IEEE 8th international symposium on applied computational intelligence and informatics (SACI)* (pp. 423–426). IEEE.

Landgrebe, T. C., & Duin, R. P. (2008). Efficient multiclass roc approximation by decomposition via confusion matrix perturbation analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *30*(5), 810–822.

Lekou, D., Mouzakis, F., Anastasopoulo, A., & Kourosis, D. (2009). Fused acoustic emission and vibration techniques for health monitoring of wind turbine gearboxes and bearings. In *European wind energy conference and exhibition, (EWEC 2009), Marseille, France* (pp. 78–82). European Wind Energy Association.

Lertampaiporn, S., Thammarongtham, C., Nukoolkit, C., Kaewkamnerdpong, B., & Ruengjitchatchawalya, M. (2013). Heterogeneous ensemble approach with discriminative features and modified-smotebagging for pre-mirna classification. *Nucleic acids research*, *41*(1), e21–e21.

Li, H., Lian, X., Guo, C., & Zhao, P. (2013). Investigation on early fault classification for rolling element bearing based on the optimal frequency band determination. *Journal of Intelligent Manufacturing*, *26*(1), 1–10.

Liao, T. W. (2008). Classification of weld flaws with imbalanced class data. *Expert Systems with Applications*, *35*(3), 1041–1052.

Liu, X.-Y., & Zhou, Z.-H. (2006). The influence of class imbalance on cost-sensitive learning: An empirical study. In *Data mining, ICDM'06. Sixth international conference on* (pp. 970–974). IEEE.

Lu, Y., Tang, J., & Luo, H. (2012). Wind turbine gearbox fault detection using multiple sensors with features level data fusion. *Journal of Engineering for Gas Turbines and Power*, *134*(4), 042501.

Modi, S., Lin, Y., Cheng, L., Yang, G., Liu, L., & Zhang, W. (2011). A socially inspired framework for human state inference using expert opinion integration. *Mechatronics, IEEE/ASME Transactions on*, *16*(5), 874–878.

Montazer, G. A., & Escalera, S., et al. (2012). Error correcting output codes for multiclass classification: Application to two image vision problems. In *2012 16th CSI international symposium on artificial intelligence and signal processing (AISP)* (pp. 508–513). IEEE.

Nadeau, C., & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, *52*(3), 239–281.

Nie, M., & Wang, L. (2013). Review of condition monitoring and fault diagnosis technologies for wind turbine gearbox. *Procedia CIRP*, *11*, 287–290.

Pazzani, M. J., Merz, C. J., Murphy, P. M., Ali, K., Hume, T., & Brunk, C. (1994). Reducing misclassification costs. In *ICML* (Vol. 94, pp. 217–225).

Provost, F., & Domingos, P. (2003). Tree induction for probability-based ranking. *Machine Learning*, *52*(3), 199–215.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann.

Rennie, J. D. (2001). *Improving multi-class text classification with naive bayes*. Ph.D. dissertation, Massachusetts Institute of Technology.

Rodríguez, J., Kuncheva, L., & Alonso, C. (2006). Rotation forest: A new classifier ensemble method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *28*(10), 1619–1630.

Salahshoor, K., Kordestani, M., & Khoshro, M. (2010). Fault detection and diagnosis of an industrial steam turbine using fusion of svm (support vector machine) and anfis (adaptive neuro-fuzzy inference system) classifiers. *Energy*, *35*(12), 5472–5482.

Samuel, P. D., & Pines, D. J. (2005). A review of vibration-based techniques for helicopter transmission diagnostics. *Journal of Sound and Vibration*, *282*(1–2), 475–508. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0022460X04003244

Sánchez, L., & Couso, I. (2012). Singular spectral analysis of ill-known signals and its application to predictive maintenance of windmills with scada records. *Soft Computing*, *16*(5), 755–768.

Santos, P., Villa, L., Reñones, A., Bustillo, A., & Maudes, J. (2012). Wind turbines fault diagnosis using ensemble classifiers. *Advances in Data Mining. Applications and Theoretical Aspects*, *7377*, 67–76.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Napolitano, A. (2010). Rusboost: A hybrid approach to alleviating class imbalance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *40*(1), 185–197.

Simani, S., & Fantuzzi, C. (2006). Dynamic system identification and model-based fault diagnosis of an industrial gas turbine prototype. *Mechatronics*, *16*(6), 341–363.

Simonoff, J. S. (1995). Smoothing categorical data. *Journal of Statistical Planning and Inference*, *47*(1), 41–69.

Soua, S., Van Lieshout, P., Perera, A., Gan, T.-H., & Bridge, B. (2013). Determination of the combined vibrational and acoustic emission signature of a wind turbine gearbox and generator shaft in service as a pre-requisite for effective condition monitoring. *Renewable Energy*, *51*, 175–181.

Stander, C., & Heyns, P. (2006) Transmission path phase compensation for gear monitoring under fluctuating load conditions. *Mechanical Systems and Signal Processing*, *20*(7), 1511–1522. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0888327005000919

Tan, A. C., Gilbert, D., & Deville, Y. (2003). Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics*, *14*, 206–217.

Teixidor, D., Grzenda, M., Bustillo, A., & Ciurana, J. (2013). Modeling pulsed laser micromachining of micro geometries using machine-learning techniques. *Journal of Intelligent Manufacturing*, 1–14. doi:10.1007/s10845-013-0835-x.

Vijayakumar, S., & Schaal, S. (2006). Approximate nearest neighbor regression in very high dimensions. In G. Shakhnarovich, T. Darrell, & P. Indyk (Eds.), *Nearest-neighbor methods in learning and vision: Theory and practice* (pp. 103–142). Cambridge, MA: MIT Press.

Villa, L. F., Reñones, A., Perán, J. R., & de Miguel, L. J. (2011). Angular resampling for vibration analysis in wind turbines under non-linear speed fluctuation. *Mechanical Systems and Signal Processing*, *25*(6), 2157–2168. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0888327011000677

Villa, L. F., Reñones, A., Perán, J. R., & de Miguel, L. J. (2012). Statistical fault diagnosis based on vibration analysis for gear test-bench under non-stationary conditions of speed and load. *Mechanical Systems and Signal Processing*, *29*, 436–446.

Wang, S., & Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In *Computational intelligence and data mining, 2009. IEEE symposium on CIDM'09* (pp. 324–331). IEEE.

Wang, J., Gao, R. X., & Yan, R. (2014). Integration of EEMD and ICA for wind turbine gearbox diagnosis. *Wind Energy*, *17*(5), 757–773.

Wang, Y.-C., Wang, X.-B., Yang, Z.-X., & Deng, N.-Y. (2010). Prediction of enzyme subfamily class via pseudo amino acid composition by incorporating the conjoint triad feature. *Protein and Peptide Letters*, *17*(11), 1441–1449.

Wang, S., & Yao, X. (2012). Multiclass imbalance problems: Analysis and potential solutions. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *42*(4), 1119–1130.

Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*, 2nd ed. Morgan Kaufmann, http://www.cs.waikato.ac.nz/ml/weka.

Yager, R. R. (2004). Owa aggregation over a continuous interval argument with applications to decision making. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *34*(5), 1952–1963.

Zhan, Y., Makis, V., & Jardine, A. K. (2006). Adaptive state detection of gearboxes under varying load conditions based on parametric modelling. *Mechanical Systems and Signal Processing*, *20*(1), 188–221. [Online]. Available http://www.sciencedirect.com/science/article/pii/S0888327004001499

Zhou, Z.-H., & Liu, X.-Y. (2010). On multi-class cost-sensitive learning. *Computational Intelligence*, *26*(3), 232–257.

Ziani, R., Felkaoui, A., & Zegadi, R. (2014). Bearing fault diagnosis using multiclass support vector machines with binary particle swarm optimization and regularized fisher's criterion. *Journal of Intelligent Manufacturing*, 1–13. doi:10.1007/s10845-014-0987-3.

Capítulo 6

Conclusiones y futuras líneas de actuación

6.1. Conclusiones

En el presente estudio se han usado técnicas de Minería de Datos para optimizar dos procesos industriales: la fabricación de piezas metálicas de geometría compleja y el diagnóstico de fallos en aerogeneradores. Se ha seguido para ello un procedimiento de validación experimental más riguroso que el de estudios anteriores, en los que no se utilizó validación cruzada estratificada ni se tuvieron en cuenta algunas particularidades de los problemas analizados, como el desequilibrio entre fallos y aciertos para el diagnóstico de fallos en aerogeneradores y la cantidad limitada de datos en el caso de fabricación de piezas metálicas de geometría compleja.

En lo referente a la fabricación de piezas metálicas de geometría compleja, se analizaron dos aspectos, al considerar tanto la etapa de obtención de la geometría como la etapa de obtención de la calidad superficial requerida. Para ello, se consideraron dos procesos: el fresado de micro-cavidades en un acero inoxidable 316L y el pulido superficial de piezas de acero de dos materiales (LaserForm^{TM} y Orvar Supreme), que habían sido fresadas previamente para tener la calidad superficial inicial típica en operaciones de pulido.

En los experimentos de fresado de micro-cavidades se desarrollaron modelos para predecir cuatro aspectos geométricos de la pieza: volumen, anchura, diámetro y longitud, y un aspecto de la productividad del proceso, la tasa de eliminación de material. Para ello,

se emplearon técnicas de regresión capaces de predecir los valores reales de las cinco variables anteriores, en función de los parámetros de control de avance del cabezal láser (intensidad, frecuencia, velocidad, tiempo) y de los valores programados de mecanizado, así como los errores entre los valores programados y los reales. A través de los regresores construidos puede predecirse cuál será la calidad de la geometría de la pieza obtenida, según sean los parámetros programados en la máquina.

Frente a los trabajos anteriores, que se centraban en el uso de redes neuronales, se han empleado más técnicas de clasificación, haciendo especial énfasis en los ensembles. Además se han justificado de forma teórica las causas del menor de error de predicción de los ensembles en base a la reducción de las componentes sesgo y varianza, comprobando experimentalmente dicha reducción. La técnica más robusta resultó Iterated Bagging con árboles de decisión M5 sin poda como regresores base, ya que no fue peor de forma estadísticamente significativa que otra técnica de regresión en ninguna de las 14 salidas consideradas. Las precisiones obtenidas para cada una de las salidas predichas, en términos del RMS en porcentaje respecto de la media del valor absoluto de la variable, varían según la salida considerada. La mayor precisión se obtiene para el diámetro medido, con un error de apenas el 2,57 %, mientras que el modelo menos preciso es del error relativo de longitud, con un RMS de hasta el 79,91 %.

En el caso del pulido superficial, la variable a predecir es la rugosidad final, definida en una escala discreta a través del estándar ISO 4288. Para ello, se desarrollan modelos de clasificación ordinal que estiman el valor de dicha variable en función las ocho entradas del proceso: material procesado, distancia focal de offset, diámetro del haz, potencia, tasa de alimentación, densidad de energía, gas auxiliar y rugosidad inicial. Al igual que en el caso de la obtención de la geometría, los modelos permiten analizar cuál será la calidad de la pieza en función de los parámetros de control de la máquina. Sólo se encontró un trabajo anterior que analizaba el problema con técnicas de Minería de Datos, pero mediante técnicas de regresión. En el presente estudio se han empleado técnicas de clasificación ordinal, obteniéndose mayor precisión. El clasificador más preciso fue un ensemble Rotation Forest con técnicas de clasificación ordinal y utilizando tanto los atributos iniciales como su discretización (83,75 % de acierto). La precisión que se obtuvo en el trabajo anterior mediante técnicas de regresión fue del 79,26 %.

En lo referente al diagnóstico de fallos en aerogeneradores, se han analizado los fallos

más típicos de este tipo de maquinaria, el desequilibrio y el desalineamiento. Para ello, se ha desarrollado un clasificador que diagnostica automáticamente a partir de variables obtenidas con un análisis de la vibraciones de la máquina si la máquina funciona correctamente o existe alguno de los fallos anteriormente citados. En primer lugar, se ha analizado el diagnóstico de fallos como un problema con equilibrio entre las clases (i.e., con un número similar de situaciones de cada fallo que de situaciones de funcionamiento correcto). El mejor método encontrado en la experimentación ha sido un SVM con kernel lineal. Otros métodos han obtenido similares porcentajes de acierto pero con tiempos más altos de entrenamiento y de ajuste. Se ha determinado que la causa de los buenos resultados del SVM con kernel lineal es que la definición de los atributos tiende a formar conjuntos linealmente separables. Además, se ha observado que no hay redundancia en los datos y que se pierde precisión si se reduce el número de variables aplicando alguna técnica de selección de variables.

Posteriormente, a diferencia de otros trabajos anteriores, se ha tenido en cuenta el desequilibrio entre las clases (i.e., el hecho de que la máquina vayan a registrarse a nivel de explotación muchas más situaciones de funcionamiento correcto que de fallo). De este modo, se ha analizado qué tipo de métrica es más adecuada para medir la precisión de los clasificadores y qué técnica soporta mejor el desequilibrio, limitando además cuál es el grado máximo de desequilibrio asumible por las técnicas de predicción. Además, en la validación experimental se han utilizado validaciones cruzadas para medir la precisión de los clasificadores, lo que ha permitido analizar si existen diferencias estadísticamente significativas entre la precisión de diferentes clasificadores mediante tests *t* remuestreados corregidos, aspecto que no había sido evaluado en trabajos anteriores. El mejor método encontrado en la experimentación fue un clasificador sensible al coste con un ensemble Rotation Forest con árboles C4.4 como clasificadores base. Cuando el porcentaje de instancias de las clases minoritarias baja hasta el rango 6,5 - 13 %, mantiene un valor alto de precisión en términos de la métrica MacroF (0,75), mientras que con la mayor parte de clasificadores no es posible obtener resultados precisos si las instancias de las clases minoritarias representan menos del 10.5 % del total.

6.2. Futuras líneas de actuación

En lo referente al diagnóstico de fallos en aerogeneradores, se plantean las siguientes líneas de actuación futura:

- Respecto de la experimentación, se desea probar con otros tipos de fallos aparte del desequilibrio y el desalineamiento.

- Respecto del procedimiento de análisis, podría modificarse el entrenamiento de los métodos de clasificación para tratar de optimizar una métrica específica de problemas desequilibrados. Como punto de partida podrían tomarse algunos estudios [109], en los que se desarrolla una técnica de Boosting que minimiza la métrica F para problemas de clasificación binarios. En el caso considerado en el presente trabajo sería necesario generalizar la función a minimizar al caso multiclase, usando otra métrica y teniendo en cuenta la distinta representación de las clases.

- Cuando se considera el desequilibro entre fallos y situaciones de funcionamiento correcto presente en un contexto real de producción, la mejor técnica de predicción encontrada combinaba dos conceptos: hacer al clasificador sensible al diferente coste de las clases según su frecuencia y proyectar los datos en espacios rotados mediante análisis de componentes principales (PCA), un sistema de proyección no supervisado (que no tiene en cuenta las clases). En este contexto podrían probarse otras combinaciones de proyecciones no supervisadas diferentes de PCA, o bien usar proyecciones supervisadas que tuvieran en cuenta el diferente coste de las clases. Cabe destacar que ciertos sistemas de proyección supervisados han sido usadas en el contexto de clasificación, pero definiendo otro objetivo para los espacios resultantes de la proyección. Por ejemplo, un análisis discriminante lineal (Linear Discriminant Analysis, LDA) consigue espacios transformados en los que la separabilidad lineal entre dos clases es máxima [170]. En este caso habría que desarrollar una proyección que tuviera en cuenta la diferencia de frecuencias entre clases.

Por otro lado, en lo que concierne a la fabricación de piezas metálicas de geometría compleja, se considera que podría ser interesante probar la validez del análisis en otros tipos de procesos que siguen el mismo estándar industrial.

Finalmente, en ambos casos sería interesante analizar el *problema inverso*. Es decir, una vez que se han establecido modelos que predicen el valor de una variable de salida a partir de las variables de entrada, tomar cómo punto de partida el nivel de salida deseado para buscar qué valores de las entradas son necesarios para conseguir dicho nivel. Por ejemplo, para el caso de fabricación de piezas metálicas de geometría compleja, analizar si hay algunas entradas que minimizan los errores respecto de todas las dimensiones de la pieza, o si no es posible optimizar simultáneamente todas las dimensiones.

Bibliografía

[1] Aggarwal, C. C. (2014). Instance-based learning: A survey. *Data Classification: Algorithms and Applications*, p. 157.

[2] Aha, D. W., Kibler, D., y Albert, M. K. (1991). Instance-based learning algorithms. *Machine learning*, 6(1):37–66.

[3] Alcorta García, E. (1992). *Modelado de sistemas mediante redes neuronales*. Tesis doctoral, Universidad Autónoma de Nuevo León.

[4] AlMana, A. M. y Aksoy, M. (2014). An overview of inductive learning algorithms. *International Journal of Computer Applications*, 88(4):20–28.

[5] Aluja Banet, T. (2001). *La minería de datos, entre la estadística y la inteligencia artificial*. Institut dÉstadística de Catalunya.

[6] Ashkenasi, D., Kaszemeikat, T., Mueller, N., Dietrich, R., Eichler, H. J., e Illing, G. (2011). Laser trepanning for industrial applications. *Physics Procedia*, 12:323–331.

[7] Balasubramanian, K., Buvanashekaran, G., y Sankaranarayanasamy, K. (2007). Mathematical and ann modeling of nd: Yag laser welding of thin ss sheets. *Journal Manufacturing Engineering*, 6(2):56–60.

[8] Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., y Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424.

[9] Barszcz, T. y Randall, R. B. (2009). Application of spectral kurtosis for detection of a tooth crack in the planetary gear of a wind turbine. *Mechanical Systems and Signal Processing*, 23(4):1352–1365.

[10] Bartelmus, W. y Zimroz, R. (2009). Vibration condition monitoring of planetary gearbox under varying external load. *Mechanical Systems and Signal Processing*, 23(1):246–257. Special Issue: Non-linear Structural Dynamics.

[11] Batista, G. E., Prati, R. C., y Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM Sigkdd Explorations Newsletter*, 6(1):20–29.

[12] Bauer, E. y Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1-2):105–139.

[13] Beale, R. y Jackson, T. (2010). *Neural Computing-an introduction*. CRC Press.

[14] Bellazzi, R. y Zupan, B. (2008). Predictive data mining in clinical medicine: current issues and guidelines. *International journal of medical informatics*, 77(2):81–97.

[15] Bentz, Y. y Merunka, D. (2000). Neural networks and the multinomial logit for brand choice modelling: a hybrid approach. *Journal of Forecasting*, 19(3):177–200.

[16] Bereznai, M., Pelsöczi, I., Tóth, Z., Turzó, K., Radnai, M., Bor, Z., y Fazekas, A. (2003). Surface modifications induced by ns and sub-ps excimer laser pulses on titanium implant material. *Biomaterials*, 24(23):4197–4203.

[17] Berry, M. J. y Linoff, G. S. (2004). *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons.

[18] Berson, A., Smith, S., y Thearling, K. (2000). *Building data mining applications for CRM*. McGraw-Hill.

[19] Bertsekas, D. P. y Tsitsiklis, J. N. (1995). Neuro-dynamic programming: an overview. En *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, volumen 1, pp. 560–564. IEEE.

[20] Bishop, Christopher M and others (2006). *Pattern recognition and machine learning*, volumen 4. springer New York.

[21] Biswas, R., Kuar, A., Sarkar, S., y Mitra, S. (2010). A parametric study of pulsed nd: Yag laser micro-drilling of gamma-titanium aluminide. *Optics & Laser Technology*, 42(1):23–31.

[22] Blanco, M. I. (2009). The economics of wind energy. *Renewable and Sustainable Energy Reviews*, 13(6):1372–1382.

[23] Boillat, E., Kolossov, S., Glardon, R., Loher, M., Saladin, D., y Levy, G. (2004). Finite element and neural network models for process optimization in selective laser sintering. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 218(6):607–614.

[24] Bordatchev, E. V., Hafiz, A. M., y Tutunea-Fatan, O. R. (2014). Performance of laser polishing in finishing of metallic surfaces. *The International Journal of Advanced Manufacturing Technology*, 73(1-4):35–52.

[25] Borghesani, P., Pennacchi, P., Chatterton, S., y Ricci, R. (2014). The velocity synchronous discrete fourier transform for order tracking in the field of rotating machinery. *Mechanical Systems and Signal Processing*, 44(1):118–133.

[26] Boser, B. E., Guyon, I. M., y Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. En *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152. ACM.

[27] Boyd, S. y Vandenberghe, L. (2009). *Convex optimization*. Cambridge university press.

[28] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

[29] Breiman, L., Friedman, J., Stone, C. J., y Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

[30] Brenes, G. S. *Comprendiendo la Estadística Inferencial*.

[31] Briscoe, G. y Caelli, T. (1996). *A compendium of Machine Learning*, volumen 1. Intellect Books.

[32] Brown, G. (2010). Ensemble learning. En *Encyclopedia of Machine Learning*, pp. 312–320. Springer.

[33] Bruneel, D., Matras, G., Le Harzic, R., Huot, N., König, K., y Audouard, E. (2010). Micromachining of metals with ultra-short ti-sapphire lasers: Prediction and optimization of the processing time. *Optics and Lasers in Engineering*, 48(3):268–271.

[34] Bühlmann, P. y Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, pp. 477–505.

[35] Bühlmann, P. y Yu, B. (2000). Discussion of additive logistic regression: A statistical view. *The Annals of Statistics*, 28:377–386.

[36] Buja, A. y Stuetzle, W. (2006). Observations on bagging. *Statistica Sinica*, 16(2):323.

[37] Bustillo, A., Díez-Pastor, J., Quintana, G., y García-Osorio, C. (2011a). Avoiding neural network fine tuning by using ensemble learning: application to ball-end milling operations. *The International Journal of Advanced Manufacturing Technology*, 57(5):521–532.

[38] Bustillo, A. y Rodríguez, J. J. (2013). Online breakage detection of multitooth tools using classifier ensembles for imbalanced data. *International Journal of Systems Science*, (ahead-of-print):1–13. doi: 10.1080/00207721.2013.775378.

[39] Bustillo, A., Ukar, E., Rodriguez, J. J., y Lamikiz, A. (2011b). Modelling of process parameters in laser polishing of steel components using ensembles of regression trees. *International Journal of Computer Integrated Manufacturing*, 24(8):735–747.

[40] Bustillo, A., Ukar, E., Rodriguez, J. J., y Lamikiz, A. (2011c). Modelling of process parameters in laser polishing of steel components using ensembles of regression trees. *International Journal of Computer Integrated Manufacturing*, 24(8):735–747.

[41] Cabezas Defaus, Neydis and Cruz Pérez, Pedro P and Iglesias Martínez, Miguel E and others (2013). El procesamiento estadístico de orden superior: herramienta útil para el análisis de señales. *Revista de Ciencia y Tecnología*, (20):30–37.

[42] Campanelli, S., Casalino, G., y Contuzzi, N. (2013). Multi-objective optimization of laser milling of 5754 aluminum alloy. *Optics & Laser Technology*, 52:48–56.

[43] Chakrabarti, Soumen and Cox, Earl and Frank, Eibe and Güting, Ralf Hartmut and Han, Jiawei and Jiang, Xia and Kamber, Micheline and Lightstone, Sam S and Nadeau, Thomas P and Neapolitan, Richard E and others (2008). *Data Mining: Know It All*. Morgan Kaufmann.

[44] Chandrasekaran, M., Muralidhar, M., Krishna, C. M., y Dixit, U. (2010). Application of soft computing techniques in machining performance prediction and optimization: a literature review. *The International Journal of Advanced Manufacturing Technology*, 46(5-8):445–464.

[45] Chang, C.-C. y Lin, C.-J. (2001). Training v-support vector classifiers: theory and algorithms. *Neural computation*, 13(9):2119–2147.

[46] Chawla, N. V., Bowyer, K. W., Hall, L. O., y Kegelmeyer, W. P. (2011). Smote: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*.

[47] Chawla, N. V., Lazarevic, A., Hall, L. O., y Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. En *Knowledge Discovery in Databases: PKDD 2003*, pp. 107–119. Springer.

[48] Chen, J. y Hao, G. (2012). Research on the fault diagnosis of wind turbine gearbox based on bayesian networks. *Practical Applications of Intelligent Systems*, pp. 217–223.

[49] Chen, T.-C. y Darling, R. B. (2008). Laser micromachining of the materials using in microfluidics by high precision pulsed near and mid-ultraviolet nd: Yag lasers. *Journal of materials processing technology*, 198(1):248–253.

[50] Cheng, J., Perrie, W., Edwardson, S., Fearon, E., Dearden, G., y Watkins, K. (2009). Effects of laser operating parameters on metals micromachining with ultrafast lasers. *Applied Surface Science*, 256(5):1514–1520.

[51] Choudhary, A. K., Harding, J. A., y Tiwari, M. K. (2009). Data mining in manufacturing: a review based on the kind of knowledge. *Journal of Intelligent Manufacturing*, 20(5):501–521.

[52] Ciurana, J., Arias, G., y Ozel, T. (2009). Neural network modeling and particle swarm optimization (pso) of process parameters in pulsed laser micromachining of hardened aisi h13 steel. *Materials and Manufacturing Processes*, 24(3):358–368.

[53] Cong-Zhong, C., Jun-Fang, P., Yu-Feng, W., Xing-Jian, Z., y Ting-Ting, X. (2009). Density prediction of selective laser sintering parts based on support vector regression. *Acta Physica Sinica*, 58:S1–S19.

[54] Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 215–242.

[55] Crammer, K. y Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292.

[56] Davies, A. (1998). *Handbook of condition monitoring: techniques and methodology*. Springer.

[57] Dekel, O., Shalev-Shwartz, S., y Singer, Y. (2003). Smooth ε-insensitive regression by loss symmetrization. En *Learning Theory and Kernel Machines*, pp. 433–447. Springer.

[58] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30.

[59] Dietterich, T. (2002). Bias-variance analysis of ensemble learning. *7th Course of the International School on Neural Networks ER Caianiello, Ensemble Methods for Learning Machines, Vietri-sul-Mare, Salerno, Italy*.

[60] Díez-Pastor, J., Bustillo, A., Quintana, G., y García-Osorio, C. (2012). Boosting projections to improve surface roughness prediction in high-torque milling operations. *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, pp. 1–11.

[61] Domingos, P. (2000). A unified bias-variance decomposition. En *Proceedings of 17th International Conference on Machine Learning. Stanford CA Morgan Kaufmann*, pp. 231–238.

[62] Dubey, A. y Yadava, V. (2008). Laser beam machining–a review. *International Journal of Machine Tools and Manufacture*, 48(6):609–628.

[63] Dubitzky, W., Granzow, M., y Berrar, D. P. (2007). *Fundamentals of data mining in genomics and proteomics*. Springer.

[64] Duda, R. O., Hart, P. E., y Stork, D. G. (1999). *Pattern classification*. John Wiley & Sons,.

[65] Ferreiro, S., Sierra, B., Irigoien, I., y Gorritxategi, E. (2011). Data mining for quality control: Burr detection in the drilling process. *Computers & Industrial Engineering*, 60(4):801–810.

[66] Frank, E. y Hall, M. (2001). *A simple approach to ordinal classification*. Springer.

[67] Frank, E., Trigg, L., Holmes, G., y Witten, I. H. (2000). Technical note: Naive bayes for regression. *Machine Learning*, 41(1):5–25.

[68] Freund, Y. y Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. En *Computational learning theory*, pp. 23–37. Springer.

[69] Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. Academic press.

[70] Fyfe, K. y Munck, E. (1997). Analysis of computed order tracking. *Mechanical Systems and Signal Processing*, 11(2):187–205.

[71] Gajjar, T. y Chauhan, N. A review on image mining frameworks and techniques. *International journal of computer science and information technologies*, 3:4064–4066.

[72] Galar, M., Fernández, A., Barrenechea, E., Bustince, H., y Herrera, F. (2012). A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(4):463–484.

[73] Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4):42–47.

[74] Ghanem, A. S., Venkatesh, S., y West, G. (2010). Multi-class pattern classification in imbalanced data. En *Pattern Recognition (ICPR), 2010 20th International Conference on*, pp. 2881–2884. IEEE.

[75] Golden, R. M. (1996). *Mathematical methods for neural network analysis and design*. MIT Press.

[76] Guo, W. (2007). Effect of irradiation parameters on morphology of polishing df2 (aisi-o1) surface by nd: Yag laser. *Advances in Materials Science and Engineering*, 2007.

[77] Gupta, A. y Lam, S. M. (1998). Weight decay backpropagation for noisy data. *Neural Networks*, 11(6):1127–1138.

[78] Gutu, I., Petre, C., Mihailescu, I., Taca, M., Alexandrescu, E., e Ivanov, I. (2002). Surface treatment with linearly polarized laser beam at oblique incidence. *Optics & Laser Technology*, 34(5):381–388.

[79] Hammer, B. y Gersmann, K. (2003). A note on the universal approximation capability of support vector machines. *Neural Processing Letters*, 17(1):43–53.

[80] Han, J., Kamber, M., y Pei, J. (2006). *Data mining: concepts and techniques*. Morgan Kaufmann.

[81] Harding, JA and Shahbaz, M and Kusiak, A and others (2006). Data mining in manufacturing: a review. *Journal of Manufacturing Science and Engineering*, 128(4):969–976.

[82] Hastie, T., Tibshirani, R., y Friedman, J. (2009). *The elements of statistical learning*, volumen 2. Springer.

[83] He, H. y Garcia, E. A. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284.

[84] Henzold, G. (2006). *Geometrical dimensioning and tolerancing for design, manufacturing and inspection: a handbook for geometrical product specification using ISO and ASME standards*. Butterworth-Heinemann.

[85] Hitz, C. B., Ewing, J. J., y Hecht, J. (2012). *Introduction to laser technology*. John Wiley & Sons.

[86] Ho, T. K. (1998). The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(8):832–844.

[87] Hoens, T. R., Qian, Q., Chawla, N. V., y Zhou, Z.-H. (2012). Building decision trees for the multi-class imbalance problem. En *Advances in Knowledge Discovery and Data Mining*, pp. 122–134. Springer.

[88] Hornik, K., Stinchcombe, M., y White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

[89] Hua, M., Shao, T., Hong, Y. T., y Man, E. C. H. (2004). Influence of pulse duration on the surface morphology of assab df-2 (aisi-01) cold work steel treated by yag laser. *Surface and Coatings Technology*, 185(2):127–136.

[90] Hua, Meng and Shao, TM and Tam, HY and others (2007). Surface transformation of df-2 steel after continuous mode laser irradiation. *Journal of materials processing technology*, 192:89–96.

[91] Huang, X., Acero, A., Hon, H.-W., y Foreword By-Reddy, R. (2001). *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall PTR.

[92] Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *Information Theory, IEEE Transactions on*, 14(1):55–63.

[93] James, G. (1998). *Majority vote classifiers: theory and applications*. Tesis doctoral, Stanford University.

[94] James, G. M. (2003). Variance and bias for general loss functions. *Machine Learning*, 51(2):115–135.

[95] Jiawei, H. y Kamber, M. (2001). Data mining: concepts and techniques. *Morgan Kaufmann*, 5.

[96] Jimin, C., Jianhua, Y., Shuai, Z., Tiechuan, Z., y Dixin, G. (2007). Parameter optimization of non-vertical laser cutting. *The International Journal of Advanced Manufacturing Technology*, 33(5-6):469–473.

[97] John, G. H. y Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. En *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pp. 338–345. Morgan Kaufmann Publishers Inc.

[98] Jones, M. T. (2008). *Artificial Intelligence A System Approach*. Laxmi Publications, Ltd.

[99] Joselin Herbert, G., Iniyan, S., Sreevalsan, E., y Rajapandian, S. (2007). A review of wind energy technologies. *Renewable and Sustainable Energy Reviews*, 11(6):1117–1145.

[100] Jurman, G. y Furlanello, C. (2010). A unifying view for performance measures in multi-class prediction. *arXiv preprint arXiv:1008.2908*.

[101] Kan, M. S., Tan, A. C., y Mathew, J. (2015). A review on prognostic techniques for non-stationary and non-linear rotating systems. *Mechanical Systems and Signal Processing*, 62:1–20.

[102] Karnakis, D., Knowles, M., Petkov, P., Dobrev, T., y Dimov, S. (2007). Surface integrity optimisation in ps-laser milling of advanced engineering materials. *Lasers in Manufacturing LIM*.

[103] Kearns, M. J. y Valiant, L. G. (1988). *Learning Boolean formulae or finite automata is as hard as factoring*. Harvard University, Center for Research in Computing Technology, Aiken Computation Laboratory.

[104] Keerthi, S. S. y Lin, C.-J. (2003). Asymptotic behaviors of support vector machines with gaussian kernel. *Neural computation*, 15(7):1667–1689.

[105] Khorram, A., Yazdi, M. S., Ghoreishi, M., y Moradi, M. (2010). Using ann approach to investigate the weld geometry of ti 6al 4v titanium alloy. *International Journal of Engineering and Technology*, 2(5):491–498.

[106] Kohavi, R. (1995). The power of decision tables. En *Machine Learning: ECML-95*, pp. 174–189. Springer.

[107] Kohavi, Ron and others (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. En *IJCAI*, volumen 14, pp. 1137–1145.

[108] Kohavi, Ron and Wolpert, David H and others (1996). Bias plus variance decomposition for zero-one loss functions. En *ICML*, pp. 275–283.

[109] Kokkinos, I. (2010). Boundary detection using f-measure-, filter-and feature-(f3) boost. En *Computer Vision–ECCV 2010*, pp. 650–663. Springer.

[110] Kotsiantis, S. B., Zaharakis, I., y Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. En *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pp. 3–24, Amsterdam, The Netherlands, The Netherlands. IOS Press.

[111] Kramer, S., Widmer, G., Pfahringer, B., y De Groeve, M. (2001). Prediction of ordinal classes using regression trees. *Fundamenta Informaticae*, 47(1):1–13.

[112] Kuncheva, L. I. (2004). *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.

[113] Kurniadi, K. A., Ryu, K., y Kim, D. (2014). Real-time parameter adjustment and fault detection of remote laser welding by using ann. *International journal of precision engineering and manufacturing*, 15(6):979–987.

[114] Kusiak, A. y Li, W. (2011). The prediction and diagnosis of wind turbine faults. *Renewable Energy*, 36(1):16–23.

[115] Lamikiz, A., Sanchez, J., Lopez de Lacalle, L., y Arana, J. (2007). Laser polishing of parts built up by selective laser sintering. *International Journal of Machine Tools and Manufacture*, 47(12):2040–2050.

[116] Le Roux, N., Bengio, Y., y Fitzgibbon, A. (2012). Improving first and second-order methods by modeling uncertainty. *Optimization for Machine Learning*, p. 403.

[117] LeCun, Y. A., Bottou, L., Orr, G. B., y Müller, K.-R. (2012). Efficient backprop. En *Neural networks: Tricks of the trade*, pp. 9–48. Springer.

[118] Lee, S.-W. (1999). *Advances in handwriting recognition*, volumen 34. World Scientific.

[119] Leuridan, J., Kopp, G. E., Moshrefi, N., y Vold, H. (1995). High resolution order tracking using kalman tracking filters-theory and applications. Technical report, SAE Technical Paper.

[120] Lin, H.-T. y Li, L. (2005). Novel distance-based svm kernels for infinite ensemble learning. En *Proceedings of the 12th International Conference on Neural Information Processing*, pp. 761–766.

[121] Luykx, K. (2012). *Bias-Variance Decomposition for model selection*. FNWI, University of Amsterdam.

[122] Mai, T. y Lim, G. (2004). Micromelting and its effects on surface topography and properties in laser polishing of stainless steel. *Journal of Laser Applications*, 16(4):221–228.

[123] Mason, L., Baxter, J., Bartlett, P., y Frean, M. (1999). Boosting algorithms as gradient descent in function space. NIPS.

[124] Mease, D., Wyner, A., y Buja, A. (2007). Cost-weighted boosting with jittering and over/under-sampling: Jous-boost. *Journal of Machine Learning Research*, 8:409–439.

[125] Meir, R. y Rätsch, G. (2003). An introduction to boosting and leveraging. En *Advanced lectures on machine learning*, pp. 118–183. Springer.

[126] Menard, S. (2002). *Applied logistic regression analysis*, volumen 106. Sage.

[127] Meng, H., Liao, J., Zhou, Y., y Zhang, Q. (2009). Laser micro-processing of cardiovascular stent with fiber laser cutting system. *Optics & Laser Technology*, 41(3):300–302.

[128] Menon, A., Mehrotra, K., Mohan, C. K., y Ranka, S. (1996). Characterization of a class of sigmoid functions with applications to neural networks. *Neural Networks*, 9(5):819–835.

[129] Milborrow, D. (2008). Generation costs rise across the board. *Wind Power Monthly*.

[130] Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine learning*, 4(2):227–243.

[131] Mitchell, T. M. (1999). Machine learning and data mining. *Communications of the ACM*, 42(11):30–36.

[132] Mohan, C. K. (2000). *Frontiers of expert systems: reasoning with limited knowledge*. Springer.

[133] Montero, F. H., Marín, E. P., Uribe, V. A., y Barrios, M. L. R. (2004). Análisis de vibraciones para el diagnóstico aplicando procesamiento estadístico de orden superior. *Ingeniería Mecánica*, 7(2):75–80.

[134] Mood, A. y Graybill, F. (1974). *Introduction to the Theory of Statistics*. McGraw-Hill.

[135] Mosquera, G., De la Victoria, M., y Armas, R. (2001). Las vibraciones mecanicas y su aplicacion al mantenimiento predictivo. *Centro de Altos Estudios Gerenciales ISID. Caracas*.

[136] Muhammad, N., Whitehead, D., Boor, A., Oppenlander, W., Liu, Z., y Li, L. (2012). Picosecond laser micromachining of nitinol and platinum–iridium alloy for coronary stent applications. *Applied Physics A*, 106(3):607–617.

[137] Nadeau, C. y Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52(3):239–281.

[138] Nilsson, N. J. (1996). Introduction to machine learning. an early draft of a proposed textbook.

[139] Nüsser, C., Wehrmann, I., y Willenborg, E. (2011). Influence of intensity distribution and pulse duration on laser micro polishing. *Physics Procedia*, 12:462–471.

[140] Orallo, J. H., Quintana, M. J. R., y Ramírez, C. F. (2004). *Introducción a la Minería de Datos*. Pearson Prentice Hall.

[141] Oza, N. C. y Tumer, K. (2008). Classifier ensembles: Select real-world applications. *Information Fusion*, 9(1):4–20.

[142] Pedrycz, W. y Chen, S.-M. (2014). *Social Networks: A Framework of Computational Intelligence*. Springer.

[143] Pham, D., Dimov, S., y Petkov, P. (2007). Laser milling of ceramic components. *International Journal of Machine Tools and Manufacture*, 47(3):618–626.

[144] Polikar, R. (2012). Ensemble learning. En *Ensemble Machine Learning*, pp. 1–34. Springer.

[145] Provost, F. y Domingos, P. (2003). Tree induction for probability-based ranking. *Machine Learning*, 52(3):199–215.

[146] Puche, R. (2008). *Nuevos Métodos de Diagnosis de Excentricidad y otras Asimetrías Rotóricas en Máquinas Eléctricas de Inducción a través del Análisis de la Corriente Estatórica*. Tesis doctoral, Tesis Doctoral], Universidad Politécnica de Valencia, 2008.[Links].

[147] Qi, H. y Lai, H. (2012). Micromachining of metals and thermal barrier coatings using a 532nm nanosecond fiber laser. *Physics Procedia*, 39:603–612.

[148] Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

[149] Quinlan, J. R. (1996). Improved use of continuous attributes in c4. 5. *arXiv preprint cs/9603103*.

[150] Quinlan, John R and others (1992). Learning with continuous classes. En *Proceedings of the 5th Australian joint Conference on Artificial Intelligence*, volumen 92, pp. 343–348. Singapore.

[151] Quintana, G., Ciurana, J. d., y Ribatallada, J. (2010). Surface roughness generation and material removal rate in ball end milling operations. *Materials and Manufacturing Processes*, 25(6):386–398.

[152] Ramos, J., Bourell, D., y Beaman, J. (2003). Surface over-melt during laser polishing of indirect-sls metal parts. En *Materials Research Society Symposium Proceedings*, pp. 53–61.

[153] Ramos, J., Murphy, J., Wood, K., Bourell, D., y Beaman, J. (2001). Surface roughness enhancement of indirect-sls metal parts by laser surface polishing. En *Solid Freeform Fabrication Proceedings*, pp. 28–38.

[154] Rätsch, G., Onoda, T., y Müller, K.-R. (2001). Soft margins for adaboost. *Machine learning*, 42(3):287–320.

[155] Re, M. y Valentini, G. (2011). *Ensemble methods: a review*. Chapman & Hall.

[156] Rojas, R. (1996). *Neural networks: a systematic introduction*. Springer Science & Business Media.

[157] Rokach, L. (2009). *Pattern classification using ensemble methods*. World Scientific.

[158] Rosset, S. y Segal, E. (2002). Boosting density estimation. En *Advances in Neural Information Processing Systems*, pp. 641–648.

[159] Rovira, C. (2007). *Estadística bàsica*, volumen 278. Edicions Universitat Barcelona.

[160] Russell, S. J. y Norvig, P. (2002). *Artificial intelligence: a modern approach (International Edition)*. Pearson US Imports & PHIPEs.

[161] Russo, R. E., Mao, X., Liu, H., Gonzalez, J., y Mao, S. S. (2002). Laser ablation in analytical chemistr–a review. *Talanta*, 57(3):425–451.

[162] Saberian, M. J. y Vasconcelos, N. (2011). Multiclass boosting: Theory and algorithms. En *Advances in Neural Information Processing Systems*, pp. 2124–2132.

[163] Saklakoglu, I. E. y Kasman, S. (2011). Investigation of micro-milling process parameters for surface roughness and milling depth. *The International Journal of Advanced Manufacturing Technology*, 54(5-8):567–578.

[164] Sammut, C. y Webb, G. I. (2011). *Encyclopedia of machine learning*. Springer.

[165] Samuel, P. D. y Pines, D. J. (2005). A review of vibration-based techniques for helicopter transmission diagnostics. *Journal of Sound and Vibration*, 282(1-2):475–508.

[166] Sánchez, L. y Couso, I. (2012). Singular spectral analysis of ill-known signals and its application to predictive maintenance of windmills with scada records. *Soft Computing*, 16(5):755–768.

[167] Santos, E. C., Shiomi, M., Osakada, K., y Laoui, T. (2006). Rapid manufacturing of metal components by laser forming. *International Journal of Machine Tools and Manufacture*, 46(12):1459–1468.

[168] Sasaki, M. y Shinnou, H. (2005). Spam detection using text clustering. En *Cyberworlds, 2005. International Conference on*, pp. 4–pp. IEEE.

[169] Schölkopf, B. (2001). The kernel trick for distances. En *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, volumen 13, p. 301. MIT Press.

[170] Schwardt, L. y du Preez, J. (2003). Manipulating feature space. *Lecture Materials. University of Stellenbosch, South Africa*.

[171] Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., y Napolitano, A. (2010). Rusboost: A hybrid approach to alleviating class imbalance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(1):185–197.

[172] Sewell, M. (2008). *Ensemble learning*. University College London Research Publications Service.

[173] Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

[174] Shawe-Taylor, J. y Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

[175] Shen, X., Yao, J., Wang, Y., y Yang, J. (2004). Density prediction of selective laser sintering parts based on artificial neural network. En *Advances in neural networks-ISNN 2004*, pp. 832–840. Springer.

[176] Simani, S. y Fantuzzi, C. (2006). Dynamic system identification and model-based fault diagnosis of an industrial gas turbine prototype. *Mechatronics*, 16(6):341–363.

[177] Simonoff, J. S. (1995). Smoothing categorical data. *Journal of Statistical Planning and Inference*, 47(1):41–69.

[178] Smola, A. J. (2000). *Advances in large margin classifiers*. MIT press.

[179] Stander, C. y Heyns, P. (2006). Transmission path phase compensation for gear monitoring under fluctuating load conditions. *Mechanical Systems and Signal Processing*, 20(7):1511–1522.

[180] Steyn, J., Naidoo, K., y Land, K. (2007). Improvement of the surface finish obtained by laser ablation with a nd: Yag laser on pre-ablated tool steel. pp. 1–5.

[181] Suen, Y. L., Melville, P., y Mooney, R. J. (2005). Combining bias and variance reduction techniques for regression trees. En *Machine Learning: ECML 2005*, pp. 741–749. Springer.

[182] Takefuji, Y. (1992). *Neural network parallel computing*. Springer.

[183] Takezawa, K. (2005). *Introduction to nonparametric regression*, volumen 606. John Wiley & Sons.

[184] Teixidor, D., Ferrer, I., Ciurana, J., y Özel, T. (2012). Optimization of process parameters for pulsed laser milling of micro-channels on aisi h13 tool steel. *Robotics and Computer-Integrated Manufacturing*.

[185] Teixidor, D., Grzenda, M., Bustillo, A., y Ciurana, J. (2013). Modeling pulsed laser micromachining of micro geometries using machine-learning techniques. *Journal of Intelligent Manufacturing*, pp. 1–14.

[186] Torabi, A. J., Joo, E. M., Xiang, L., Siong, L. B., Lianyin, Z., Jie, P. S., Junhong, Z., Lin, S., Sheng, H., y Tijo, J. T. T. (2009). A survey on artificial intelligence technologies in modeling of high speed end-milling processes. En *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on*, pp. 320–325. IEEE.

[187] Tricarico, L., Sorgente, D., y Scintilla, L. D. (2011). Experimental investigation on fiber laser cutting of ti6al4v thin sheet. *Advanced Materials Research*, 264:1281–1286.

[188] Vachtsevanos, G., Lewis, F., Roemer, M., Hess, A., y Wu, B. (2006). *Intelligent fault diagnosis and prognosis for engineering systems, 2006*. John Wiley.

[189] Valentini, G. y Dietterich, T. G. (2003). Low bias bagged support vector machines. En *ICML*, pp. 752–759.

[190] Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

[191] Vapnik, V. (2000). *The nature of statistical learning theory*. Springer.

[192] Vapnik, V. N. y Vapnik, V. (1998). *Statistical learning theory*, volumen 1. Wiley New York.

[193] Vázquez, E., Amaro, A., Ciurana, J., y Rodríguez, C. A. (2015). Process planning considerations for micromilling of mould cavities used in ultrasonic moulding technology. *Precision Engineering*, 39:252–260.

[194] Villa, L. F., Reñones, A., Perán, J. R., y De Miguel, L. J. (2011). Angular resampling for vibration analysis in wind turbines under non-linear speed fluctuation. *Mechanical Systems and Signal Processing*, 25(6):2157–2168.

[195] Villa, L. F., Reñones, A., Perán, J. R., y de Miguel, L. J. (2012). Statistical fault diagnosis based on vibration analysis for gear test-bench under non-stationary conditions of speed and load. *Mechanical Systems and Signal Processing*, 29:436–446.

[196] Wang, G.-W., Zhang, C., y Zhuang, J. (2012). An application of classifier combination methods in hand gesture recognition. *Mathematical Problems in Engineering*, 2012.

[197] Wang, J., Gao, R. X., y Yan, R. (2014). Integration of EEMD and ICA for wind turbine gearbox diagnosis. *Wind Energy*, 17(5):757–773.

[198] Wang, S. y Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. En *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on*, pp. 324–331. IEEE.

[199] Webb, G. I. (2000). Multiboosting: A technique for combining boosting and wagging. *Machine learning*, 40(2):159–196.

[200] Weiss, M. L. (2007). *Neuronal Network Research Horizons*. Nova Publishers.

[201] Wenyi, L., Zhenfeng, W., Jiguang, H., y Guangfeng, W. (2013). Wind turbine fault diagnosis method based on diagonal spectrum and clustering binary tree SVM. *Renewable Energy*, 50:1–6.

[202] Whitehouse, D. J. (2004). *Surfaces and their Measurement*. Elsevier.

[203] Witten, I. H., Frank, E., y Mark, A. (2011). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

[204] Woolley, E. M. (2013). *Elements of classic parametric estimation*.

[205] Matej, H. y Gómez M. E. (2013). Conciencia. capítulo 2, la inteligencia artificial. http://www.ugr.es/ setchift/docs/conciencia_capitulo_2.pdf.

[206] Yilbas, B., Akhtar, S., y Karatas, C. (2011). Laser trepanning of a small diameter hole in titanium alloy: Temperature and stress fields. *Journal of Materials Processing Technology*, 211(7):1296–1304.

[207] Yongxin, F. (2012). Study of fault diagnosis method for wind turbine with decision classification algorithms and expert system. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 10(5):905–910.

[208] Young, T. Y. (1994). *Handbook of pattern recognition and image processing (vol. 2): computer vision*. Academic Press, Inc.

[209] Yousef, B. F., Knopf, G. K., Bordatchev, E. V., y Nikumb, S. K. (2003). Neural network modeling and analysis of the material removal process during laser machining. *The International Journal of Advanced Manufacturing Technology*, 22(1-2):41–53.

[210] Yu, H., Huang, X., Hu, X., y Cai, H. (2010). A comparative study on data mining algorithms for individual credit risk evaluation. En *Management of e-Commerce and e-Government (ICMeCG), 2010 Fourth International Conference on*, pp. 35–38. IEEE.

[211] Yuan, X. y Abouelenien, M. (2012). A boosting method for learning from uneven data for improved face recognition. En *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volumen 2, pp. 119–122. IEEE.

[212] Zhan, Y., Makis, V., y Jardine, A. K. (2006). Adaptive state detection of gearboxes under varying load conditions based on parametric modelling. *Mechanical Systems and Signal Processing*, 20(1):188–221.

[213] Zhang, Z., Liu, D., Dai, G., y Jordan, M. I. (2012). Coherence functions with applications in large-margin classification methods. *The Journal of Machine Learning Research*, 13(1):2705–2734.

[214] Zhao, G. Y., Yan, Y., Zhao, C. Z., Wang, C., y Zhang, H. (2013). Data mining in load forecasting of power system. En *Proceedings of the International Conference on Information Engineering and Applications (IEA) 2012*, pp. 769–777. Springer.

[215] Zhao, L., Mammadov, M., y Yearwood, J. (2010). From convex to nonconvex: A loss function analysis for binary classification. En *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pp. 1281–1288. IEEE.

[216] Zhou, Z.-H. y Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *Knowledge and Data Engineering, IEEE Transactions on*, 18(1):63–77.